Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Aleš Papler

Uporaba tehnologije RFID in šifriranja za zaščito izdelkov in ugotavljanje ponaredkov

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Mira Trebar

Ljubljana, 2018

Zahvala

Zahvaljujem se svoji mentorici doc. dr. Miri Trebar za vso strokovno pomoč, nasvete in usmerjanje pri izdelavi magistrskega dela. Hvaležen sem tudi družini, ki me je v času študija spodbujala in mi stala ob strani.

Aleš Papler, 2018

Kazalo

Povzetek

Abstract

1 Uvod **1**

1.1 Motivacija 1

1.2 Predlagana rešitev 2

1.3 Prispevki 2

1.4 Struktura dela 3

2 Pregled področja **5**

2.1 Tehnologija RFID 5

2.2 Tehnologija NFC 8

2.3 Zagotavljanje avtentičnosti 16

2.4 Kriptografija in kriptografski algoritmi 22

3 Zasnova in načrtovanje prototipa sistema **31**

3.1 Ideja 31

3.2 Aktivnosti sistema 35

3.3 Model 1 38

3.4 Model 2 43

3.5 Model 3 48

4 Razvoj sistema **55**

4.1 Mobilna aplikacija 55

4.2 Aplikacijski strežnik in podatkovna baza 63
4.3 Razvojno okolje 63
4.4 Mobilne in spletne tehnologije 65

5 Rezultati testiranja in analiza **71**
5.1 Vpis nove značke v sistem oskrbovalne verige 72
5.2 Dodajanje novega zapisa gibanja značke v oskrbovalno verigo . 76
5.3 Preverjanje avtentičnosti značke 77
5.4 Delovanje aplikacije – časovne zahteve 83
5.5 Primeri uporabe 87
5.6 Možni napadi 89
5.7 Analiza SWOT 91

6 Sklepne ugotovitve **95**

Seznam uporabljenih kratic

kratica	angleško	slovensko
AES	Advanced Encryption Standard	simetrični kriptosistem, naslednik DES
API	Application Programming Interface	aplikacijski programski vmesnik
DES	Data Encryption Standard	simetrični (bločni) kriptosistem
EPC	Electronic Product Code	elektronska koda izdelka
GDSN	Global Data Synchronisation Network	globalno omrežje za sinhronizacijo podatkov
GS1	Global Standards One	globalna nepridobitna organizacija GS1
HTTP	HyperText Transfer Protocol	protokol za izmenjavo vsebin na spletu
HTTPS	HyperText Transfer Protocol Secure Sockets	protokol, ki omogoča varno spletno povezavo
JSON	JavaScript Object Notation	notacija objektov JavaScript
NFC	Near Field Communication	komunikacija kratkega dosega
REST	Representational State Transfer	predstavitev stanj prenosa spletnih zahtev
RFID	Radio-Frequency Identification	radiofrekvenčna identifikacija
RSA	Rivest-Shamir-Adleman	kriptosistem z javnimi ključi
SHA	Secure Hash Algorithms	varni zgoščevalni algoritmi
SWOT	Strengths, Weaknesses, Opportunities and Threats	prednosti, slabosti, priložnosti in nevarnosti
SQL	Structured Query Language	strukturiran povpraševalni jezik

Povzetek

Naslov: Uporaba tehnologije RFID in šifriranja za zaščito izdelkov in ugotavljanje ponaredkov

Na trgu se iz leta v leto pojavlja vedno več ponarejenih izdelkov. Razlikovanje med originalnimi izdelki in ponaredki je vse težje. Ob hitrem razvoju in dostopnosti sodobnih tehnologij se zato poraja vprašanje, ali je mogoče z njihovo uporabo zagotoviti avtentičnost izdelkov in na ta način lažje zaznati in preprečiti razpečevanje ponarejenih izdelkov. Raziskali smo nove pristope za zaščito avtentičnosti izdelkov in nove mehanizme, ki otežujejo ponarejanje. Zasnovali smo prototip sistema, ki temelji na oskrbovalni verigi in uporablja tehnologijo radiofrekvenčne identifikacije (RFID) za označevanje izdelkov. Šifrirni algoritmi (DES, AES, RSA) so uporabljeni za zaščito proti kopiranju podatkov na značkah. Definirali smo tri modele z različnimi stopnjami zaščite. Izdelali smo mobilno aplikacijo, ki potrošniku pred nakupom izdelka, opremljenega z značko komunikacije kratkega dosega (NFC), omogoča preverjanje njegove avtentičnosti z uporabo pametnega telefona.

Ključne besede

šifriranje, avtentikacija, ponaredek, izdelek, RFID, NFC

Abstract

Title: Product authentication and counterfeit detection using RFID technology and cryptography

Each year the consumer market turns up more and more counterfeit items. Differentiating between verified original products and counterfeits is becoming increasingly difficult. However, the fast development and wide accessibility of modern technology both pose the question whether their utilization may ensure the authenticity of products, all the while increasing the capability of detecting and preventing sales of counterfeits. We have researched new approaches of protecting the authenticity of products, and new mechanisms which make falsification of products more difficult. We have devised a prototype of a system which is based on the supply chain and uses Radio Frequency Identification (RFID) to label products. Encryption algorithms (DES, AES, RSA) have been utilized to prevent product-tag copying. We have defined three models, all with different protection levels. Additionally, we have developed a mobile application which enables the consumer to review a product's authenticity using its Near Field Communication (NFC) tag, should the product in question have one, with their mobile phone.

Keywords

encryption, authentication, counterfeit, product, RFID, NFC

Poglavje 1

Uvod

1.1 Motivacija

Na svetovnem trgu se vsako leto pojavlja čedalje več ponarejenih izdelkov in izdelkov, ki kršijo pravice intelektualne lastnine. Po raziskavah Organizacije za gospodarsko sodelovanje in razvoj (OECD) naj bi bila njihova vrednost enaka 200 milijardam ameriških dolarjev [1], kar ima močan negativen vpliv na celotno gospodarsko panogo. Ponarejevalci so v preteklosti večinoma ponarejali ure in oblačila višjega cenovnega razreda, danes pa lahko med ponaredki zasledimo tudi prehranske in farmacevtske izdelke. Ti imajo pogosto negativen vpliv na zdravje končnih potrošnikov [2]. Z rastjo spletne prodaje in odpiranja tujih trgov se je povečeval tudi trg ponaredkov.

Razlikovanje med originalnimi izdelki in ponaredki postaja vse težje. Proti ponaredkom se oblasti bojujejo z ozaveščanjem potrošnikov, izpeljavo pravnih postopkov in sankcij za ponarejevalce ter z mednarodnim povezovanjem in sodelovanjem svetovnih organizacij. Lastniki blagovnih znamk pa nenehno iščejo izboljšane in stroškovno upravičene načine pakiranja in označevanja izdelkov, s katerimi bi ponudili potrošnikom možnost, da pred nakupom preverijo izvor in avtentičnost izdelkov.

1.2 Predlagana rešitev

Na trgu obstajata dva načina zaščite proti ponaredkom. Prvi je vpeljava tehnologije, s katero je na podlagi fizičnih značilnosti izdelka možno ugotoviti, ali je originalen. Drugi način je sistem sledljivosti, ki omogoča pregled zgodovine izdelka v oskrbovalni verigi. Čeprav sta oba načina že dolgo poznana, se v praksi še nista dobro uveljavila. Izjema so le farmacevtski izdelki z dobro razvito sledljivostjo vse od proizvodnje do končnih uporabnikov.

V magistrskem delu smo se ukvarjali s problemom zaščite izdelkov v oskrbovalni verigi. Zasnovali smo varen sistem, ki bo z uporabo šifrirnih algoritmov omogočal zaščito proti kopiranju podatkov na značkah radiofrekvenčne identifikacije (RFID – angl. *Radio-Frequency Identification).* Sistem bo sestavljen iz štirih delov – značk RFID, mobilne aplikacije, spletnega strežnika in podatkovne baze. Za določanje natančne lokacijske in časovne sledljivosti izdelkov, ki bodo opremljeni z značkami RFID, bomo sistem vključili v oskrbovalno verigo. S preverjanjem avtentičnosti in sledljivosti značk RFID bomo skušali nedvoumno razlikovati med originalnimi in ponarejenimi izdelki.

1.3 Prispevki

Predvideni prispevki magistrske naloge so:

- Načrtovanje varnega modela za zaščito podatkov, ki se shranjujejo na značke RFID.
- Zaščita in zagotavljanje avtentičnosti izdelka, opremljenega z značko RFID, ter enolično preverjanje avtentičnosti in odkrivanje ponaredkov.
- Rezultat zasnove in načrtovanja je sistem z mobilno aplikacijo, ki uporablja komunikacijo kratkega dosega (NFC – angl. *Near Field Communication*) in s katero lahko potrošnik s svojim pametnim telefonom v trgovini preveri avtentičnost izdelka.
- Implementacija in analiza delovanja treh konkretnih modelov z različnimi stopnjami zaščite.

1.4 Struktura dela

V Poglavju 2 bomo na kratko predstavili načine zaščite izdelkov in naredili pregled sorodnih del. Sledil bo opis tehnologij RFID in NFC ter pametnih kartic, na koncu poglavja bo sledil opis kriptografije in kriptografskih algoritmov. V Poglavju 3 bomo opisali teoretično zasnovo in načrtovanje sistema za zaščito izdelkov. Razvoj sistema ter pregled izbranih tehnologij in orodij, ki jih bomo uporabili pri načrtovanju in razvoju prototipa, sta predstavljena v Poglavju 4. V Poglavju 5 bomo zapisali rezultate testiranja in analize razvitih rešitev ter opisali primere uporabe in možne napade na sistem. Poglavje 6 bo vsebovalo sklepne misli in opis možnih izboljšav.

Poglavje 2

Pregled področja

2.1 Tehnologija RFID

Radiofrekvenčna identifikacija (angl. *Radio-Frequency Identification*) je brezžična tehnologija [3], ki za prenos podatkov med bralnikom RFID in značko RFID uporablja radiofrekvenčne valove. Začetki segajo že v 19. stoletje, ko je Faraday odkril koncept medsebojne indukcije,[1] ki je kasneje postala osnova za napajanje integriranih vezij v kratkem dosegu. V prvi polovici 20. stoletja je tehnološki napredek radiofrekvenčne tehnologije že omogočal napajanje vezij v širšem dosegu, ki so jih med drugo svetovno vojno uporabili za prisluškovalno napravo. Leta 1983 je Charles Walton uradno pridobil prvi patent za radiofrekvenčno napravo in od takrat naprej se komunikacija med dvema napravama, ki uporabljata medsebojno indukcijo, imenuje tehnologija RFID.

RFID je v zadnjih dveh desetletjih pridobila na veljavi, zlasti po tem, ko sta jo v svojih oskrbovalnih verigah pričela uporabljati ameriško obrambno ministrstvo in trgovska veriga Walmart. Glavne prednosti tehnologije RFID v primerjavi s črtnimi kodami so:

- branje značk RFID poteka z večje razdalje, četudi značke niso v vidnem

[1] Medsebojna indukcija je pojav, pri katerem sprememba električnega toka v eni tuljavi inducira napetost v drugi, bližnji tuljavi.

polju bralnika RFID,

- bralnik RFID je v primerjavi z optičnim bralnikom in črtnimi kodami zmožen identificirati več značk v enakem časovnem obdobju,
- shranjevanje več informacij o izdelku (ne le nek enolični identifikator izdelka), saj imajo značke RFID dodaten pomnilniški prostor,
- podatki, zapisani na znački RFID, se lahko aktivno spreminjajo med delovanjem.

Frekvenčna območja

Frekvenčno območje delovanja predstavlja pomemben faktor v tehnologiji RFID, saj se z njegovim spreminjanjem spremenijo tudi karakteristike elektromagnetnega valovanja [4]. Frekvenčna območja razdelimo v tri kategorije:

- nizke frekvence (angl. *low frequency*, LF),
- visoke frekvence (angl. *high frequency*, HF) in
- ultra visoke frekvence (angl. *ultra high frequency*, UHF).

Pri nizkofrekvenčnem RFID se najpogosteje uporabljata frekvenci 125 kHz in 134,2 kHz. Prednost nizkih frekvenc je v manjšem odboju signala od kovinskih in vodnih površin. Prav zaradi te lastnosti se nizkofrekvenčen RFID najpogosteje uporablja v avtomobilski in orodjarski industriji, ravnanju s kovinskimi zabojniki ter pri označevanju živine. Pokrivno območje je od nekaj centimetrov do največ dveh metrov. Slaba lastnost je ta, da hrup drugih naprav lahko moti komunikacijo med bralnikom značk in značkami. Prav tako je hitrost prenosa podatkov med napravami nizka, kar posledično onemogoča hkratno komunikacijo med bralnikom in več značkami.

Visokofrekvenčen RFID deluje pri frekvenci 13,56 MHz. Visoka frekvenca omogoča, da signal dobro prodre skozi večino materialov, vključno z vodo in telesnim tkivom. Območje delovanja je do enega metra. V primerjavi z nizkimi frekvencami je ta tehnologija cenejša ter omogoča hkratno komunikacijo

več naprav. V območju visokofrekvenčnega RFID se je uveljavila tehnologija komunikacije kratkega dosega NFC (angl. *Near Field Communication*). Z integracijo v mobilne telefone je postala razširjena in se najpogosteje uporablja tudi pri kreditnih karticah, karticah dostopa in potnih listih.

Frekvenčni pas delovanja ultra visokofrekvenčnega RFID je od 300 MHz do 3 GHz. Najpogosteje se uporabljata frekvenci 433 MHz in 2,45 GHz ter pas med 860 do 956 MHz. Je novejša tehnologija od ostalih dveh, območje delovanja je do deset metrov. Z uporabo aktivnih značk se območje delovanja poveča do dvesto metrov. Ima tudi zmožnost hkratnega identificiranja tisoč značk z uporabo zaščite proti koliziji podatkov.

Napajanje značk

Glede na vir energije, ki napaja integrirano vezje značk, se značke delijo v naslednje tri kategorije:

- pasivna značka RFID,
- aktivna značka RFID in
- polaktivna značka RFID.

Pasivne značke RFID nimajo lastnega napajanja, saj vso energijo pridobijo iz elektromagnetnega signala bralnika. Signal, ki ga prejmejo, uporabijo za napajanje svojega integriranega vezja in za pošiljanje odgovora. Zaradi te omejitve napajanja se območje delovanja zoži na nekaj centimetrov. Omejitev predstavlja tudi integrirano vezje značke, saj to ne sme biti kompleksno in energijsko potratno.

Aktivne značke RFID za napajanje integriranega vezja in pošiljanje odgovora uporabljajo lastno baterijo. Zmogljivost in kompleksnost integriranega vezja je v primerjavi s pasivnimi značkami lahko precej večja. Območje delovanja je v kombinaciji z ultra visokofrekvenčnim RFID do nekaj sto metrov.

Polaktivne značke RFID imajo vgrajeno baterijo, s katero napajajo integrirano vezje. Za pošiljanje odgovora bralniku uporabljajo energijo iz elektromagnetnega valovanja prejetega signala – enako, kot to naredijo pasivne

značke. V primerjavi z aktivnimi značkami je čas, ki je potreben, da značka preide iz stanja mirovanja v aktivno stanje, precej daljši pri polaktivni znački.

2.2 Tehnologija NFC

Za zaščito občutljiviih informacij, ki se prenašajo pri uporabi radiofrekvenčne tehnologije, sta se podjetji Philips in Sony združili pri novem projektu in leta 2002 predstavili tehnologijo NFC (angl. *Near Field Communication*, NFC) [5]. Tehnologija NFC je komunikacija kratkega dosega [6], ki za komunikacijo med dvema napravama uporablja inducirano magnetno polje. Njen obseg delovanja je z namenom zaščite informacij omejen na nekaj centimetrov. Organizacija Ecma International je konec leta 2002 razglasila NFC kot standard (ECMA-340), Mednarodna komisija za elektrotehniko (angl. *International Electrotechnical Commission*) pa je to storila leto kasneje (ISO/IEC 18092).

V letu 2004 so podjetja Nokia, Philips in Sony ustanovila neprofitno industrijsko združenje, imenovano NFC Forum, z namenom promoviranja tehnologije NFC na mobilnih napravah in osebnih računalnikih ter zagotavljanja skladnosti med proizvajalci naprav s tehnologijo NFC in proizvajalci značk NFC.

Tipi značk NFC

V združenju NFC Forum so definirali štiri tipe značk NFC in njihove specifikacije [5, 7], ki temeljijo na brezstični komunikaciji in so prosto dostopni vsem. Vsi tipi temeljijo na mednarodno sprejetih standardih.

NFC forum tip 1 temelji na standardu ISO/IEC 14443A, značke NFC so lahko namenjene le branju (angl. *read-only*) ali pa branju in pisanju. Velikost pomnilnika je do 2 kB, kar zadošča za zapis spletnega naslova ali drugih krajših tekstovnih zapisov. Prenos podatkov med bralnikom NFC in značko

poteka pri hitrosti 106 kb/s. Tip 1 ne vsebuje zaščite proti koliziji podatkov[2] in posledično omogoča le eno istočasno komunikacijo. Prednost tega tipa je preprostost integriranega vezja in posledično nižji stroški za izdelavo. Vsebina značke se lahko trajno zaščiti z 32-bitnim geslom.

NFC forum tip 2 je izpeljanka tipa 1, od njega se razlikuje le pri vgrajeni zaščiti proti koliziji podatkov, vse druge lastnosti so enake. Običajno se tipa značk 1 in 2 uporabljata pri preprostih aplikacijah, ki na pomnilniku značke ne zahtevajo velike količine zapisanih podatkov in ne potrebujejo višje stopnje varnosti.

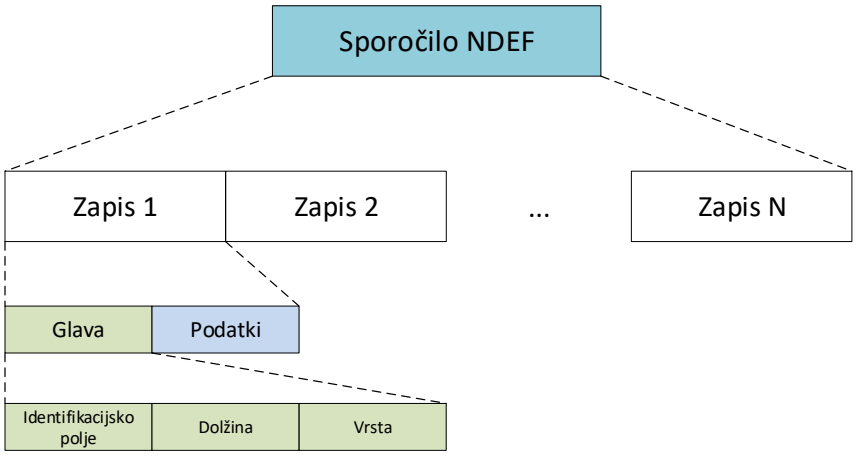
NFC forum tip 3 temelji na japonskem standardu JIS X 6319-4, ki je poznan kot FeliCa. Značka je lahko namenjena le branju ali branju in pisanju, kar se določi ob izdelavi. Velikost pomnilnika je do največ 2 MB. Vgrajeno ima zaščito proti koliziji podatkov, hitrost prenosa podatkov je 212 oziroma pri novejših izvedbah 424 kb/s. Običajno se značke tipa 3 uporablja za zahtevnejše aplikacije, njihova cena je v primerjavi z ostalimi tipi značk višja.

NFC forum tip 4 temelji na standardu ISO/IEC 14443A in ISO/IEC 14443B. Značke so namenjene le branju ali branju in pisanju, kar se določi ob izdelavi. Velikost pomnilnika je do 32 kB, prenos podatkov med značko in napravo poteka pri hitrostih 106, 212 ali 424 kb/s. Tip 4 vsebuje zaščito proti koliziji podatkov. Namenjen je zahtevnejšim aplikacijam, ki zahtevajo višjo stopnjo varnosti, kot so na primer brezkontaktne kreditne in plačilne kartice ter kartice dostopa.

[2]Kolizija podatkov je pojav, ki se zgodi, kadar bralnik iz več virov hkrati prejme podatke in jim ne more določiti njihovega izvora. Za preprečevanje tega stanja so razvili algoritme, ki se uporabljajo v zaščitah proti koliziji podatkov.

Oblika zapisa podatkov

Združenje NFC Forum je definiralo tudi standard za izmenjavo informacij med dvema napravama NFC ali med napravo in značko NFC [8]. Imenuje se NDEF (angl. *NFC Data Exchange Format*) in opredeljuje obliko in pravila kratkega binarnega sporočila, ki se uporablja pri izmenjavi in zapisu podatkov, pri tem pa se k sporočilu ne doda veliko režijskih podatkov (angl. *overhead data*). Razlog za vpeljavo sporočil NDEF je zmožnost pošiljanja podatkov kateregakoli tipa in velikosti preko enotnega vmesnika, saj se pri tem ohrani oblika sporočila. Slika 2.1 prikazuje sporočilo NDEF, ki je sestavljeno iz enega ali več zapisov. En zapis je lahko dolg do $2^{32} - 1$ B, omejitev števila zapisov v sporočilu je definirana glede na tip značke in aplikacijo, ki bere in piše sporočila NDEF.



Slika 2.1: Oblika zapisa sporočila NDEF

Glava zapisa NDEF vsebuje 3-bitno polje imenovano TNF (angl. *Type Name Format*), ki določa enega izmed osmih tipov zapisa:

- Empty,
- NFC Forum Well-Known Type,
- Media Type,

- Absolute URI,
- NFC Forum External Type,
- Unknown,
- Unchanged,
- Reserved.

Med njimi je najbolj uporaben tip *NFC Forum well-known type*, ki se deli na podtipe; pametni poster (angl. *smart poster*), golo besedilo (angl. *plain text*), enotni identifikator vira (angl. *uniform resource identifier*, URI) ter digitalni podpis.

Pametne kartice in značke NFC

Pametne kartice (angl. *smart cards*) so kartice, ki imajo vgrajeno vhodno-izhodno enoto in integrirano vezje z mikroprocesorjem ter pomnilnikom, njihova velikost in oblika sta enaki običajnim kreditnim karticam (fizične lastnosti kartice predpisuje standard ISO/IEC 7810) [5]. Razvite so bile v 70. letih, v širšo uporabo so prišle v 80. letih kot kartice za telefonske govorilnice. Kasneje so razvili pametne kartice z mikroprocesorjem in jih v 90. letih pričeli uporabljati kot debetne kartice. Največji preboj je predstavljal izum pametnih kartic v obliki kartice SIM, ki se še danes uporabljajo pri mobilnih telefonih GSM.

Leta 1993 so se podjetja Europay, MasterCard in Visa združila pri izdelavi novih specifikacij za pametne kartice z namenom, da bi jih kasneje uporabili pri svojih debetnih in kreditnih karticah. Leto kasneje so izdali prvo verzijo tehničnega standarda EMV (angl. *Europay, MasterCard, and Visa*), ki je pripomogel k standardizaciji kreditnih kartic in plačilnih terminalov. Standard EMV je osnovan na dveh standardih – ISO/IEC 7816 za stične in ISO/IEC 14443 za brezstične kartice.

Velikokrat so kot pametne kartice napačno poimenovane vse kartice, ki imajo sposobnost shranjevanja in zapisovanja podatkov. Take kartice so

dejansko pomnilniške kartice. Pametne kartice poleg branja in zapisovanja podatkov omogočajo še dodatno obdelavo podatkov, saj imajo vgrajen mikroprocesor, ki je zmožen izvajati kompleksne podatkovne operacije.

Glede na vrsto komunikacije med bralnikom pametnih kartic in pametno kartico ločimo tri tipe:

- **stične pametne kartice** (angl. *contact smart cards*) – bralnik pametnih kartic dostopa do vsebine kartice preko fizičnega kontakta (magnetni trak, čip kartice SIM),

- **brezstične (brezkontaktne) pametne kartice** (angl. *contactless smart cards*) – bralnik pametnih kartic dostopa do vsebine kartice preko oddaljenega brezstičnega vmesnika RF,

- **hibridni model** – kombinacija stične in brezstične pametne kartice. Vsebuje dve neodvisni integrirani vezji, eno za stični in eno za brezstični vmesnik.

Pametne kartice največkrat zasledimo na področju financ v obliki elektronske denarnice ter debetnih in kreditnih kartic. Najdemo jih v vseh mobilnih napravah GSM kot kartice SIM. Drugi primeri uporabe pametnih kartic so še identifikacijske kartice in kartice za nadzor dostopa, kartice zvestobe in zdravstvene kartice. V logistiki in prevozu se pametne kartice uporabljajo za vozniška dovoljenja, elektronsko cestninjenje ter kot vozovnice za javni promet.

Na trgu brezstičnih pametnih kartic je najbolj razširjen proizvajalec NXP Semiconductors z družino kartic MIFARE. Po njihovi oceni naj bi več kot 80 odstotkov brezstičnih vozovnic v prometu 750 svetovnih mest temeljilo na karticah MIFARE [9]. Družino MIFARE delimo na štiri serije [10], ki so zaradi dobre razširjenosti in dostopnosti pametnih kartic in različnih značk NFC podrobneje opisane v nadaljevanju. Opisana je tudi družina NTAG istega proizvajalca, ki predstavlja najpreprostejši tip značk NFC.

MIFARE Classic

Prva v družini kartic MIFARE je nastala serija MIFARE Classic [11], ki je v osnovi namenjena le kot medij za shranjevanje podatkov. Zaradi cenovne dostopnosti in zanesljivosti delovanja je ta serija postala priljubljena pri nadzoru dostopa, karticah zvestobe ter brezstičnih vozovnicah in vstopnicah. MIFARE Classic je na voljo v dveh izvedbah pomnilnika (1 kB ali 4 kB). Zapisani podatki na kartici so lahko zaščiteni s šifrirnim algoritmom Crypto-1, ki temelji na tokovni šifri z 48-bitnim ključem. Algoritem je bil kmalu razbit [12], zato proizvajalec svetuje uporabo drugih, varnejših serij kartic pri varnostno kritičnih aplikacijah.

MIFARE Plus

Serija brezstičnih kartic MIFARE Plus je neposreden naslednik serije Classic [13, 14]. Za zaščito dostopa še vedno ponuja algoritem Crypto-1 (za potrebe združljivosti s serijo Classic), dodatno pa še AES z 128-bitnim ključem. V serijo Plus sodijo izvedbe S, SE, X in EV1, ki se razlikujejo predvsem v velikosti pomnilnika (od 1 kB do 4 kB).

MIFARE Ultralight

V serijo Ultralight spadajo tri izvedbe – Ultralight Nano [15], Ultralight EV1 [16] in Ultralight C [17]. Vsem je skupna nizka cena in razmeroma majhen pomnilnik od 40 B do 144 B. Zaradi teh dveh lastnosti je serija Ultralight primerna za večje naklade kartic zvestobe ter brezstičnih vozovnic in vstopnic za enkratno uporabo. Izvedba Nano ne ponuja dodatne zaščite pred branjem in/ali pisanjem, EV1 ponuja zaščito z 32-bitnim geslom pred pisanjem, izvedba C pa s šifrirnim algoritmom 3DES ponuja zaščito pred branjem in pisanjem.

Ključne lastnosti kartice MIFARE Ultralight C:

- kartica ima 7-bajtni UID,
- njeno območje delovanja je do 10 centimetrov,

- hitrost prenosa podatkov med kartico in bralnikom je 106 kb/s,
- uporabniku je na voljo 144 B pomnilnika, razdeljenega v 36 strani (angl. *pages*) po 32 bitov,
- zagotavlja obstojnost zapisanih podatkov do 5 let,
- zagotavlja do 10.000 ciklov pisanja,
- zagotavlja integriteto podatkov pri prenosu z uporabo 16-bitnega algoritma CRC,
- ima vgrajeno strojno kriptografsko procesorsko enoto za šifriranje z algoritmom DES.

MIFARE DESFire

Celotna serija DESFire že v osnovi ponuja zaščito za dostop do podatkov z uporabo simetričnih šifrirnih algoritmov DES in AES [18, 19]. Velikost pomnilnika kartice je od 256 B do 8 kB. Pri vsaki kartici izvedbe DESFire EV1 ali EV2 je možno uporabiti šifriranje DES, 3DES (z dvema ali tremi ključi) ali AES. Slednji uporablja ključ dolžine 128 bitov, pri šifriranju DES je ključ lahko dolg 168 bitov (v primeru šifriranja 3DES in tremi 56-bitnimi ključi). Kartica omogoča izbiro enega izmed treh tipov komunikacije, ki poteka med kartico in bralnikom NFC. Podatki se lahko prenašajo v čistopisu, v čistopisu z dodano oznako za overjanje (angl. *message authentication code*) ali v šifrirani obliki. Pri tem je algoritem, ki se uporablja za šifriranje podatkov, enak algoritmu, ki se uporablja pri avtentikaciji.

Posebnost kartice DESFire je v aplikacijskem sistemu, ki pomnilnik kartice razdeli na med seboj neodvisne aplikacije. Pri izvedbi EV1 je število aplikacij do največ 28, pri izvedbi EV2 to število ni omejeno in je odvisno le od prostega pomnilnika. Vsaki aplikaciji se lahko dodeli od enega do štirinajst različnih ključev za dostop – to so aplikacijski ključi (angl. *application keys*). Kartica ima en glavni ključ (angl. *master key*), s katerim se upravlja kartico in njene aplikacije, vendar z njim ni možno brati vsebine aplikacij. Vsaka

aplikacija lahko vsebuje do 32 datotek, njihova največja velikost je odvisna od prostega pomnilnika značke.

Ključne lastnosti kartice MIFARE DESFire EV1:

- kartica ima 7-bajtni UID,
- predstavlja brezstično multiaplikacijsko integrirano vezje (IC),
- njeno območje delovanja je do 10 centimetrov,
- hitrost prenosa podatkov med kartico in bralnikom je 848 kb/s,
- zagotavlja obstojnost zapisanih podatkov do 10 let,
- zagotavlja do 500.000 ciklov pisanja,
- zagotavlja integriteto podatkov pri prenosu z uporabo 32-bitnega algoritma CRC,
- ima vgrajeno strojno kriptografsko procesorsko enoto za šifriranje DES in AES,
- ima vgrajen samodejni razveljavitveni mehanizem (angl. *rollback mechanism*), ki preprečuje okvaro datotečnega sistema pri njegovem upravljanju.

NTAG

Značke družine NTAG neposredno sicer ne sodijo v družino MIFARE, vendar predstavljajo najpogostejšo in najcenejšo izvedbo brezstičnih značk NFC [20, 21]. Temeljijo na standardu ISO/IEC 14443A in izpolnjujejo specifikacijo NFC forum tip 2. Trenutno so aktualne serije značk NTAG21x, ki se med seboj predvsem razlikujejo po velikosti pomnilnika od 48 B (izvedba NTAG210) do 1912 B (NTAG I2C plus 2K). Med njimi je najbolj razširjena izvedba NTAG213, njene ključne lastnosti pa so:

- značka ima 7-bajtni UID,
- njeno območje delovanja je do 10 centimetrov,
- hitrost prenosa podatkov med kartico in bralnikom je 106 kb/s,

- uporabniku je na voljo 144 B pomnilnika, razdeljenega v 36 strani (angl. *pages*) po 32 bitov,
- ima možnost uporabe 32-bitnega gesla za zaščito pomnilnika pred branjem in pisanjem,
- zagotavlja obstojnost zapisanih podatkov do 10 let,
- zagotavlja do 100.000 ciklov pisanja,
- zagotavlja integriteto podatkov pri prenosu z uporabo 16-bitnega algoritma CRC.

2.3 Zagotavljanje avtentičnosti

Na trgu obstajajo rešitve, ki se z različnimi načini pakiranja in označevanja izdelkov bojujejo proti ponaredkom [22]. Njihove skupne značilnosti lahko povzamemo v štirih točkah: (1) težko jih je podvojiti oziroma ponarediti, (2) prostemu očesu so vidne brez posebnih naprav, (3) težko jih je ponovno uporabiti in (4) vsak poskus prirejanja je opazen in ga ni moč razveljaviti.

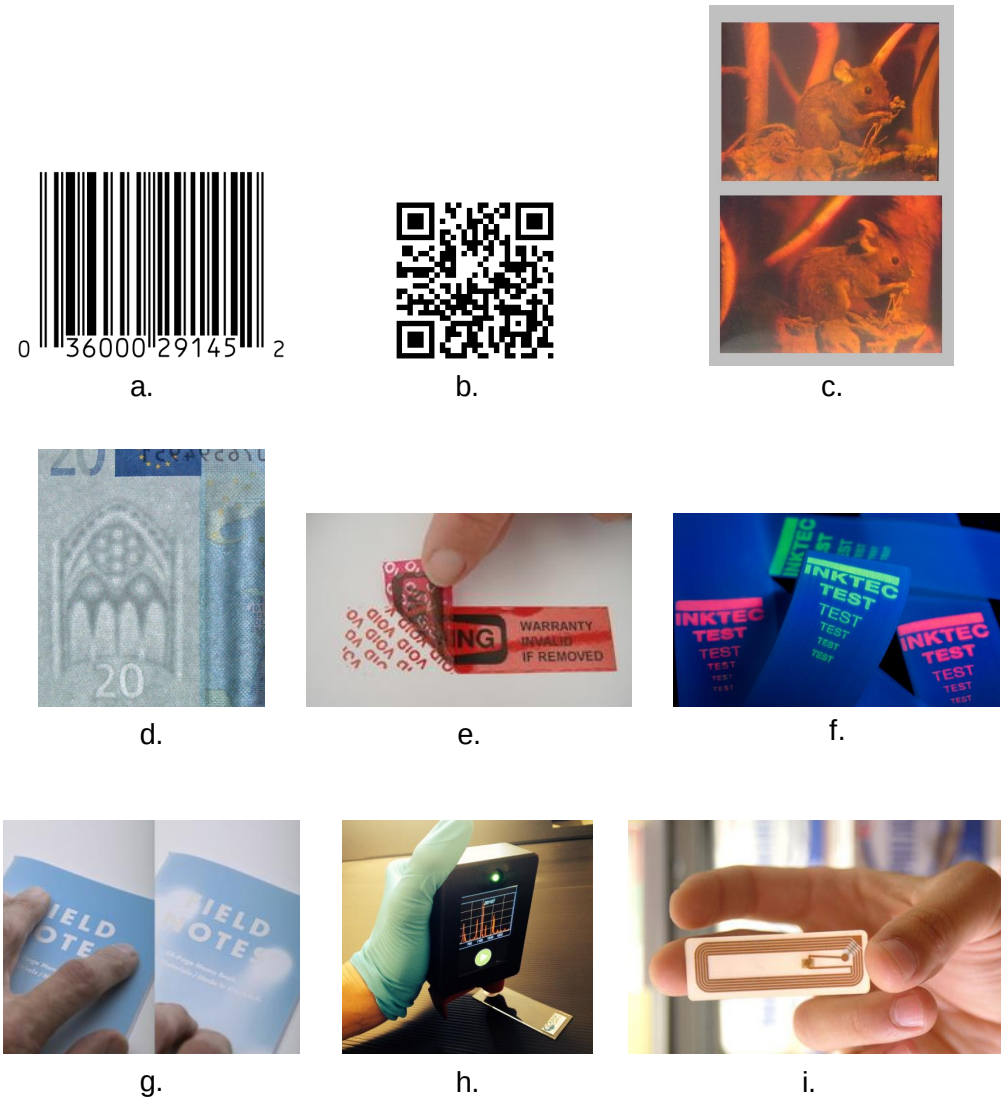
Rešitve, ki vključujejo zgoraj naštete značilnosti, delimo na štiri glavne kategorije [23]:

1. **odkrite tehnologije** – črtne kode, QR-kode (angl. *quick response*), hologrami, vodni žigi in pečati pristnosti (angl. *authentication seals*). Na Sliki 2.2 so od a. do e. prikazani njihovi primeri.

2. **prikrite tehnologije** – varnostna črnila, ultravijoličen in termokromni tisk ter digitalni vodni žigi (Slika 2.2.f. in 2.2.g.).

3. **tehnologije s forenzičnimi značilnosti** – kemične ali biološke oznake, ki reagirajo le v stiku z določenimi reagenti in jih ni moč odkriti z običajnimi analizami (Slika 2.2.h).

4. **sistemi sledljivosti** – označevanje izdelkov (črtne kode, značke RFID) in beleženje aktivnosti, ki so povezane z njimi (Slika 2.2.i.).



Slika 2.2: Med odkrite tehnologije spadajo a. črtna koda, b. koda QR, c. hologram, d. vodni tisk in e. pečat pristnosti; prikrite tehnologije obsegajo f. ultravijoličen in g. termokromni tisk; med tehnologije s forenzičnimi značilnosti spadajo h. kemične oznake; v kategorijo sistemov sledljivosti spada i. značka RFID.

Sistem sledljivosti je z uporabo značk RFID robusten in primeren za preprečevanje ponarejanj. Ta pristop varuje celotno oskrbovalno verigo pred

ponaredki in krajami, obenem pa omogoča natančno sledljivosti vseh izdelkov znotraj verige ter njihovo avtentikacijo.

2.3.1 Sistem sledljivosti v oskrbovalni verigi

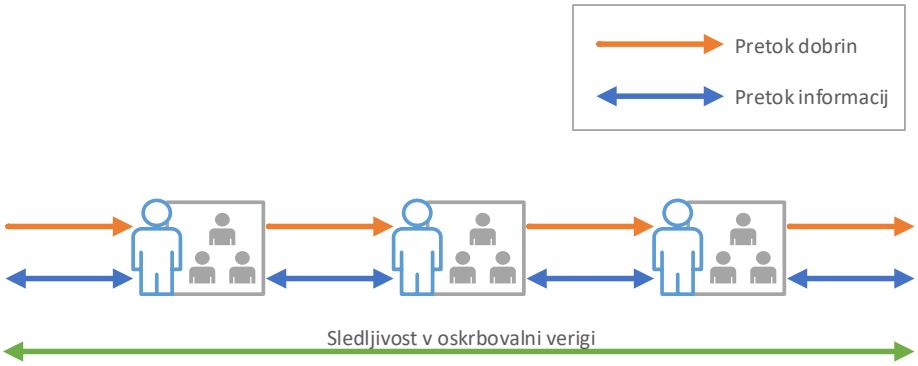
Prva rešitev celovitega sistema sledljivosti, ki temelji na tehnologiji RFID in elektronski kodi izdelka EPC[3] (angl. *Electronic Product Code*), je bila predlagana že leta 2003 in kasneje sprejeta kot priporočena praksa s strani Ameriškega vladnega urada za zdravila in prehrano (angl. *U.S. FDA*) [24]. Osnovni koncept je zasnovan na dodajanju značk RFID na vsako paleto in kartonsko embalažo, pri čemer ima vsaka značka elektronsko zapisano unikatno kodo EPC. Vse to omogoča popolno časovno in lokacijsko sledljivost znotraj farmacevtske oskrbovalne verige ter obenem zmanjša možnosti za ponarejanje in kraje.

Oskrbovalna veriga (angl. *supply chain*), poznana tudi kot dobavna, preskrbovalna oziroma oskrbna veriga, je mreža zvez in distribucijskih možnosti, ki opravljajo funkcijo nabave materialov in preoblikovanja le-teh v vmesne in končne proizvode ter funkcijo distribucije končnih proizvodov kupcem [25].

Oskrbovalna veriga se pojavlja v storitvenih ali blagovnih dejavnostih in združuje organizacije, podjetja, njihove zaposlene ter procese, zadolžene za premik blaga od dobaviteljev do končnih kupcev. Največkrat jo povezujemo s sistemom sledljivosti, saj v povezavi z njim dosežemo časovno, lokacijsko in izvorno sledljivost vsakega izdelka oziroma storitve. Za izvajanje sledljivosti je treba zagotoviti povezavo med fizičnim tokom dobrin in tokom podatkov, ki se nanašajo na njih. Med vsemi partnerji oziroma deležniki v oskrbovalni verigi poteka enosmeren pretok dobrin in dvosmeren pretok informacij, kot je prikazano na Sliki 2.3. Sledljivost poteka vzdolž celotne oskrbovalne verige.

Z namenom lažjega celovitega sporazumevanja in enotne izmenjave informacije med udeleženci oskrbovalne verige se je v praksi dobro uveljavil

[3]EPC oziroma elektronska koda izdelka je 96-bitna koda, ki enolično označuje fizičen predmet. Kodo na zahtevo dodeli neprofitna organizacija GS1, ki je zadolžena za določanje standardov ter dodeljevanje in upravljanje s kodami EPC v oskrbovalnih verigah.



Slika 2.3: Pretok dobrin in informacij ter sledljivost v oskrbovalni verigi

standard GS1 [26], ki obsega tri glavne sklope:

- Identifikacija na podlagi številke GTIN ali kode SSCC. GTIN je globalna trgovinska številka izdelka (angl. *Global Trade Item Number*), ki se uporablja za enolično identifikacijo prodajnih enot, SSCC pa je zaporedna koda zabojnika (angl. *Serial Shipping Container Code*), ki se uporablja za enolično identifikacijo logističnih oziroma skladiščnih enot.

- Označevanje prodajnih enot in zajem podatkov na podlagi črtne kode ali EPC/RFID. Z uporabo črtnih kod (te so lahko tipa EAN/UPC, QR kode, GS1-128 ali ITF-14) se zajem podatkov popolnoma avtomatizira in s tem zmanjša verjetnost napake, z uporabo RFID se postopek zajema podatkov pohitri (v primerjavi z zajemom z uporabo črtnih kod).

- Izmenjava podatkov s pomočjo elektronske izmenjave dokumentov (angl. *Electronic Data Interchange*, krajše EDI) ali z uporabo globalnega omrežja za sinhronizacijo podatkov GS1 GDSN (angl. *Global Data Synchronization Network*). GDSN omogoča vsem udeležencem oskrbovalne verige, da sinhronizirajo podatke o izdelkih in pakiranjih preko ene skupne točke. Spremembe, narejene v podatkovni zbirki enega podjetja,

se samodejno posodobijo v centralnem katalogu in so nato na voljo vsem udeležencem. Vsak udeleženec ima stalen dostop do centralnega kataloga.

2.3.2 Obstoječe rešitve

Novejši koncepti so osnovno rešitev sistema sledljivosti nadgradili z zlivanjem podatkov – izdelek označen z značko RFID pri vsakem prehodu v oskrbovalni verigi dobi dodaten zapis, opremljen s časovnim žigom (angl. *timestamp*), in s tem tvori integriteto izdelka [27]. Vsak tak zapis se zapiše v centralno podatkovno bazo proizvajalca. Obstaja tudi rešitev s porazdeljeno podatkovno bazo, kjer ima vsak deležnik svojo [28]. Pri preverjanju celotne zgodovine se izvede poizvedba po vseh porazdeljenih bazah. Na ta način lahko vsakemu izdelku natančno določimo poreklo in mu sledimo od izvora (izvor v oskrbovalni verigi predstavljajo dobavitelji) vse do cilja (končni uporabnik oziroma kupec). Sprotno preverjanje porekla izdelka je omogočeno vsem deležnikom v oskrbovalni verigi. Vsak deležnik naj bi ob prevzemu izdelka preveril njegovo poreklo in ga ob nepopolni ali dvomljivi preteklosti zavrnil. Kupec ima na prodajnem mestu možnost preveriti njegovo poreklo z bralnikom RFID. Sistem sledljivosti, ki temelji na tehnologiji RFID, naj bi bil po mnenju avtorjev zaradi dražje začetne investicije namenjen izdelkom višjega cenovnega razreda.

Ker podatki na značkah RFID niso zaščiteni pred nedovoljenim pisanjem ali kopiranjem, so raziskovalci želeli sistem sledljivosti nadgraditi še z dodatno zaščito. Značke RFID so opremili s fizično neponovljivimi funkcijami (angl. *physical unclonable function* oziroma krajše PUF), ki se nahajajo na integriranemu vezju [29]. PUF je funkcija, ki preslika izziv (angl. *challenge*) v odgovor (angl. *response*) in se nahaja v zaprtem delu nekega fizičnega objekta. Funkcijo je enostavno izvesti, vendar praktično nemogoče ugotoviti njene lastnosti le na podlagi izzivov in odgovorov. Vsak fizičen poskus poseganja v enoto PUF povzroči njeno uničenje in uničenje celotnega integriranega vezja značke. Za vsako značko so izvedli določeno število izzivov

in jih nato skupaj s pripadajočimi odgovori podpisali s skrivnim ključem izdajatelja. V podatkovno bazo so shranili izzive in podpise. Ob preverjanju pristnosti značke se znački pošlje enega izmed izzivov in nato preveri ujemanje s shranjenimi podatki. Če je podpis veljaven, to pomeni, da je značka res avtentična.

Protokol izziv – odgovor med bralnikom in značko RFID za preverjanje avtentičnosti ni najbolj varen [30]. Za doseganje višje stopnje varnosti mora biti dinamičen – značka mora na enak, ponavljajoč izziv vsakič odgovoriti drugače in ne enako, statično. V tem primeru se je mogoče izogniti napadu s prisluškovanjem. Za dodatno zaščito so avtorji želeli vsebino in komunikacijo med bralnikom in značko zaščititi še s šifriranjem. Ker simetrični šifrirni algoritmi zahtevajo, da je skrivni ključ shranjen na znački, prav tako mora biti vsa komunikacija med strežnikom, bralnikom in značko šifrirana, so avtorji želeli to izboljšati. Uporabili so asimetričen šifrirni algoritem (eliptične krivulje), saj ne zahteva shranjenega skrivnega ključa na znački. Slaba stran te rešitve je v precejšni računski zahtevnosti, ki jo mora poleg bralnika izvesti tudi značka.

Novejša izboljšava omenjenega koncepta je ohranila asimetričen šifrirni sistem eliptičnih krivulj za preverjanje avtentičnosti, vendar so avtorji večino računskih in energijsko zahtevnih metod prenesli iz pametne kartice RFID na bralnik kartic [31]. Pametna kartica RFID je imela integrirano vezje z mikroprocesorjem, ki je zadolžen za avtentikacijo med bralnikom in pametno kartico.

Kasneje so rešitvi, ki temelji na tehnologiji RFID ter simetričnih in asimetričnih kriptografskih algoritmih za avtentikacijo, dodali sistem sledljivosti v oskrbovalni verigi [32]. Vsak deležnik v oskrbovalni verigi doda svoj šifriran zapis na značko RFID, pri šifriranju se uporabi simetrično šifriranje. V vsakem zapisu je prisotna informacija o trenutnem deležniku, časovni žig prihoda in odhoda izdelka iz obrata ter informacija o naslednjem deležniku. Avtentikacijo značke so implementirali z uporabo digitalnega podpisa, ki temelji na lastno razviti shemi RSA 32. Vsakemu deležniku v verigi je poznan

njegov zaseben simetričen in asimetričen ključ ter javen asimetričen ključ naslednika. Vse skupaj tvori popolno sledljivost značk RFID v oskrbovalni verigi ter odkrivanje ponaredkov v skladiščih ali med prevozom.

2.4 Kriptografija in kriptografski algoritmi

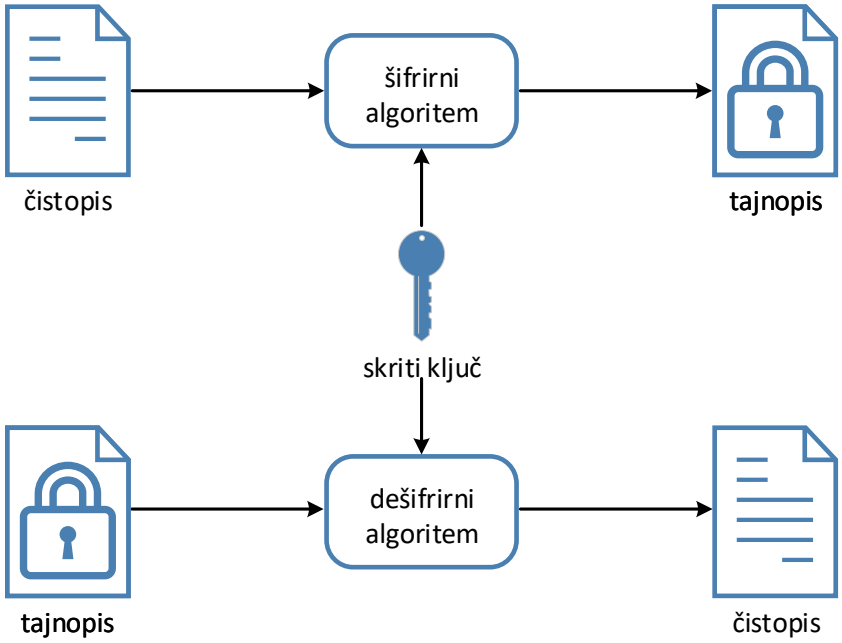
Kriptografija je veda o varni komunikaciji med dvema osebama z zavedanjem prisotnosti tretje osebe, imenovane aktivni napadalec. S pomočjo kriptografije dosežemo zaupnost (angl. *confidentiality*), celovitost (angl. *data integrity*) ter overjanje podatkov (angl. *data authentication*), identifikacijo (angl. *identification*) in preprečevanje tajenja (angl. *non-repudiation*) [33]. Kriptografske algoritme v splošnem delimo na štiri glavna področja [34]:

- klasična kriptografija – zamenjalne, bločne in tokovne šifre;
- simetrični kriptosistemi – DES in AES;
- asimetrični kriptosistemi – RSA, ElGamal ter algoritmi, ki temeljijo na eliptičnih krivuljah;
- zgoščevalne funkcije – družini funkcij SHA in MD.

2.4.1 Simetrični kriptosistemi

Simetrični kriptosistemi uporabljajo enak ključ za šifriranje in dešifriranje podatkov. Algoritma sta enaka, običajno je dešifriranje obraten postopek šifriranja (Slika 2.4). Prav zaradi teh dveh lastnosti mora biti šifrirni algoritem (posledično tudi dešifrirni) močan, skriti ključ pa mora ostati tajen – le pošiljatelj in prejemnik ga poznata.

Simetrični kriptosistemi so osnovani na substituciji, permutaciji ali na obeh – hibridna zasnova. Substitucija pomeni menjavo dela čistopisa z delom abecede, permutacija povzroči medsebojno menjavo delov čistopisa. Hibridna shema je sestavljena iz obeh funkcij, ki se izvedeta ena za drugo. Najbolj znana predstavnika simetričnih kriptosistemov sta DES in AES.



Slika 2.4: Simetričen kriptosistem

DES

DES (angl. *Data Encryption Standard*) je simetričen kriptosistem, ki spada v družino bločnih šifer [35]. V sedemdesetih letih ga je razvilo podjetje IBM v sodelovanju z Ameriškim nacionalnim inštitutom za standarde in tehnologijo (angl. *National Institute of Standards and Technology*), leta 1976 je DES uradno postal standarden kriptografski algoritem v ZDA.

Algoritem za DES razdeli čistopis v bloke dolžine 64 bitov, enako dolg je tudi ključ za šifriranje in dešifriranje (običajno se 8 bitov ključa uporabi za preverjanje paritete, moč ključa je v tem primeru enaka 56 bitom). Šifriranje z DES poteka v šestnajstih ciklih, kjer se v vsakem ciklu nad blokom izvede substitucija (menjava nekaterih bitov v bloku), nato permutacija (menjava vrstnega reda bitov v bloku) in na koncu še transformacija (združevanje bitov s ključem z operacijo ekskluzivni ali XOR). Čeprav se na prvi pogled zdi algoritem kompleksen, je njegovo izvajanje preprosto in dovolj ponavljajoče,

da je primerno za izvedbo v manjših integriranih vezjih.

Ker se je računska moč hitro povečevala, so raziskovalci želeli razviti močnejši šifrirni algoritem, kot je DES s 56-bitnim ključem. Najprej so hoteli le povečati dolžino ključa, vendar so naleteli na težavo, saj je algoritem za DES deloval le s statično določeno dolžino ključa. Naslednja ideja je bila dvojno šifriranje oziroma 2DES (angl. *double DES* oziroma 2DES) z dvema različnima ključema. Vsako sporočilo so najprej šifrirali s prvim in nato še z drugim ključem. Postopek formalno zapišemo z enačbo (2.1), kjer k_1 in k_2 predstavljata prvi in drugi ključ, m sporočilo, ki ga želimo šifrirati, z E označimo šifriranje in s C končno šifrirano sporočilo.

$$C = E(k_2, E(k_1, m)) \tag{2.1}$$

V teoriji naj bi to prispevalo k dodatni varnosti, vendar sta Merkle in Hellman kmalu dokazala, da je 2DES enako varen oziroma teoretično ekvivalenten DES s 57-bitnim ključem [36].

Varnost kriptosistema DES so nato povečali z uporabo trojnega DES (angl. *triple DES* oziroma 3DES). Sporočilo so najprej šifrirali s prvim ključem, ga dešifrirali z drugim ključem in ga na koncu ponovno šifrirali s tretjim ključem, kar zapišemo s formulo (2.2).

$$C = E(k_3, D(k_2, E(k_1, m))) \tag{2.2}$$

Tak način šifriranja zagotavlja varnost ekvivalentno ključu dolžine 168 bitov in se danes uporablja pri avtentikaciji plačilnih in pametnih kartic ter plačilnih terminalov – mednarodni standard EMV.

Obstaja tudi različica 3DES, kjer sta prvi in tretji ključ enaka, posledično se varnost zniža na ekvivalenten ključ dolžine 112 bitov, vendar se uporaba 3DES z dvema ključema za šifriranje in dešifiranje po letu 2015 odsvetuje [37].

AES

Leta 1997 je Ameriški nacionalni inštitut za standarde in tehnologijo objavil razpis za nov simetričen kriptosistem, poimenovan AES (angl. *Advanced*

Encryption Standard), ki naj bi postal naslednik DES [35]. V razpisu je bilo zahtevano, da naj bo algoritem brezplačno prosto dostopen vsem in da naj deluje nad bloki dolžine 128 bitov, dolžine ključa naj bodo 128, 192 ali 256 bitne. Leta 2000 so med vsemi predlogi izbrali algoritem *Rijndael* za najprimernejšega in ga leto kasneje razglasili za nov šifrirni standard.

AES je iterativna šifra, ki deluje nad bloki dolžine 128 bitov in dovoljuje uporabo ključev dolžine 128, 192 ali 256 bitov. Pri uporabi ključa dolžine 128 bitov je potrebnih 10 krogov (angl. *cycles*), pri 192-bitnem ključu 12 in pri 256-bitnem ključu 14 krogov.

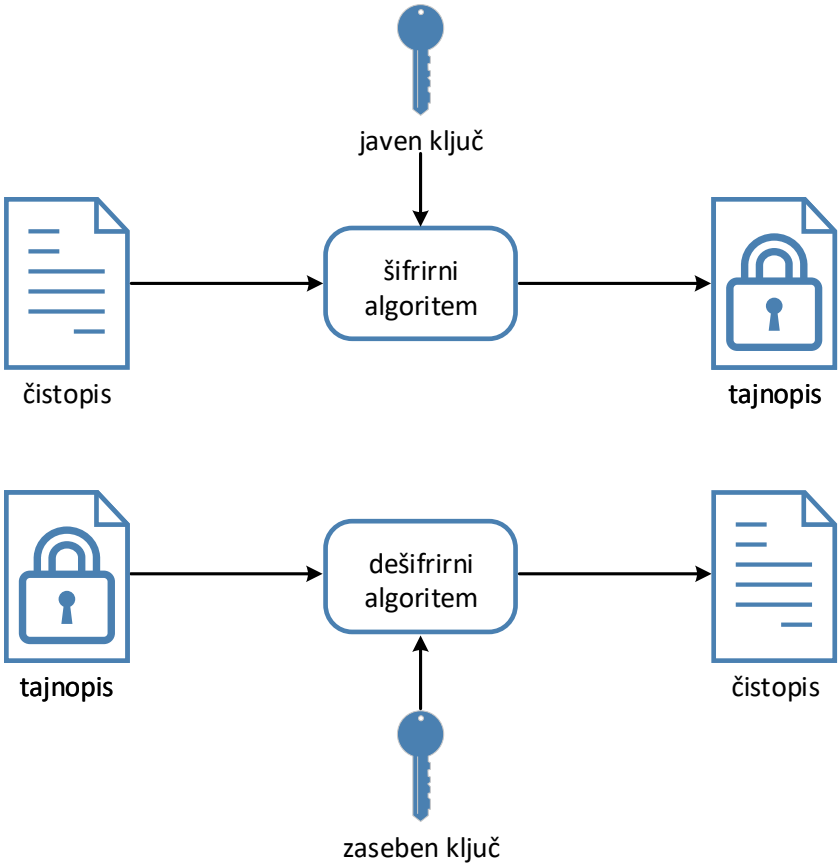
V Tabeli 2.1 je prikazana primerjava osnovnih lastnosti simetričnih kriptosistemov.

	DES	3DES	AES
tip šifre	simetrična bločna šifra	simetrična bločna šifra	simetrična bločna šifra
leto razvoja	1977	1978	1999
velikost bloka	64 bitov	64 bitov	128 bitov
dolžina ključa	56 bitov	112 bitov (2 ključa), 168 bitov (3 ključi)	128, 192, ali 256 bitov
način delovanja	Feistelova šifra	Feistelova šifra	substitucije in permutacije
število operacij	16 ciklov	16 ciklov	10, 12, 14 krogov (glede na ključ)
varnost	šibka	dokazana pomanjkljivost	velja za varno
hitrost	hitrejši kot 3DES, počasnejši kot AES	počasnejši kot DES in AES	hitrejši kot DES in 3DES

Tabela 2.1: Primerjava simetričnih kriptosistemov DES, 3DES in AES

2.4.2 Asimetrični kriptosistemi

Asimetrični kriptosistemi, poznani tudi pod pojmom kriptografija javnega ključa, uporabljajo dva različna ključa za šifriranje in dešifriranje podatkov [34, 35]. Za šifriranje se uporablja prejemnikov javni ključ, ki je javno dostopen in poznan vsem, dešifriranje lahko izvede le prejemnik s svojim skritim (zasebnim) ključem. Čeprav sta ključa v matematičnem smislu povezana, nam poznavanje javnega ključa ne razkrije nobenih informacij o skritem ključu. Postopka šifriranja in dešifriranja sta različna in ne obratna, kot je to pri simetričnih kriptosistemih. Shema šifriranja in dešifriranja je prikazana na Sliki 2.5.



Slika 2.5: Asimetričen kriptosistem

Kriptosistemi z javnimi ključi imajo pred sistemi s skritimi ključi prednost, saj za izmenjavo ključev ne potrebujejo varnega komunikacijskega kanala. Primer enega izmed protokolov za izmenjavo ključev po ne varnem kanalu je dogovor o ključu Diffie-Hellman (angl. *Diffie-Hellman key exchange*). Druga prednost asimetričnih kriptosistemov je v številu ključev pri komunikaciji z več osebami. Če želi oseba šifrirano komunicirati z več udeleženci, mora pri simetričnih kriptosistemih poznati in deliti skriti ključ z vsakim od udeležencev. Pri asimetričnih kriptosistemih zadošča, da vsi udeleženci, ki želijo komunicirati z neko osebo, poznajo le njen javen ključ.

Asimetričen kriptosistem delimo na dve področji, ki se razlikujeta glede uporabe ključev pri šifriranju in dešifriranju; pri **kriptosistemih z javnimi ključi** pošiljatelj za šifriranje čistopisa uporabi prejemnikov javni ključ, tako lahko le prejemnik dešifrira prejeto sporočilo s svojim skritim ključem; pri **digitalnih podpisih** pošiljatelj podpiše (šifrira) sporočilo s svojim zasebnim ključem, nato lahko prejemnik, ki pozna pošiljateljev javni ključ, preveri (dešifrira) sporočilo.

Slabost asimetričnih sistemov je v počasnejšem izvajanju – šifriranje in dešifriranje trajata dlje kot pri simetričnih sistemih. Dolžina ključev pri asimetričnih sistemih je daljša od dolžine ključev pri simetričnih sistemih, pri primerljivi stopnji varnosti.

Najpomembnejša kriptografska sistema z uporabo javnih ključev sta RSA in kriptografija eliptičnih krivulj.

RSA

Prvi kriptosistem z javnimi ključi RSA je bil predstavljen leta 1978 in je poimenovan po začetnicah priimkov izumiteljev – *Rivest, Shamir* in *Adleman*. Temelji na problemu faktorizacije celih števil, ki do danes ostaja nerešen in tako predstavlja trdno osnovo za varen kriptosistem. V preteklosti so bile izvedene tudi številne raziskave in analize o varnosti sistema RSA in vse do danes ni bilo ugotovljenih hujših varnostnih pomanjkljivosti, zato sistem RSA (z visoko stopnjo zaupanja) velja za varnega.

Eliptične krivulje

Kriptografija eliptičnih krivulj (angl. *Elliptic Curve Cryptography*, v nadaljevanju ECC) temelji na problemu računanja logaritmov v končnih obsegih in sta jo leta 1985 odkrita Victor Miller in Neil Koblitz. ECC predstavlja alternativo RSA, saj je za primerljivo stopnjo varnosti ključ ECC krajši od ključa RSA. Razlog za odkritje in uporabo ECC kot naslednika RSA je bilo plačilo licenčnine za uporabo algoritma RSA vse do leta 2000, dokler patent ni prišel v javno rabo.

2.4.3 Zgoščevalne funkcije

Zgoščevalne funkcije (angl. *hash functions)* so enosmerne funkcije, s katerimi se sporočilo poljubne dolžine stisne na vrednost fiksne dolžine [38]. To vrednost imenujemo zgostitev oziroma izvleček (angl. *hash value*). Vsaka zgoščevalna funkcija ima dve lastnosti, in sicer, da je iz zgostitve računsko nemogoče izračunati prvotno vrednost sporočila ter da ji ni moč poiskati trčenj. Trčenje pri zgoščevalni funkciji predstavlja dogodek, ko se dve različni sporočili zgostita v isto vrednost.

Poseben razred zgoščevalnih funkcij predstavljajo kriptografske zgoščevalne funkcije [39], ki poleg lastnosti, naštetih zgoraj, zadostujejo še naslednjim trem pogojem:

- **Odpornost na prasliko** (angl. *pre-image resistance*): pri dani zgostitvi je v doglednem času računsko neizvedljivo poiskati izvorno sporočilo.
- **Odpornost na 2. prasliko** (angl. *second pre-image resistance*): pri danem izvornem sporočilu in njegovi zgostitvi je v doglednem času računsko neizvedljivo poiskati drugo sporočilo, ki bi imelo enako zgostitev.
- **Odpornost na trke** (angl. *collision resistance*): v doglednem času je računsko neizvedljivo poiskati dve različni sporočili, ki bi imeli enako zgostitev.

Kriptografske zgoščevalne funkcije se uporabljajo pri preverjanju celovitosti in istovetnosti podatkov ter pri digitalnih podpisih. Med njimi so najbolj razširjene MD4 in MD5 (MD pomeni okrajšavo za angl. *message digest*) ter SHA (angl. *secure hash algorithm*), ki predstavlja celo družino funkcij (SHA-0, SHA-1, SHA-2 in SHA-3). Ker so v preteklosti zgoščevalne funkcije MD4, MD5 ter SHA-0 in SHA-1 že bile razbite, se danes za izračun zgostitev priporoča uporabo funkcij iz družine SHA-2 ali SHA-3. V praksi najbolj uporabljena zgoščevalna funkcija je SHA-256, ki spada v družino SHA-2 in izračuna 256-bitno zgostitev.

Poglavje 3

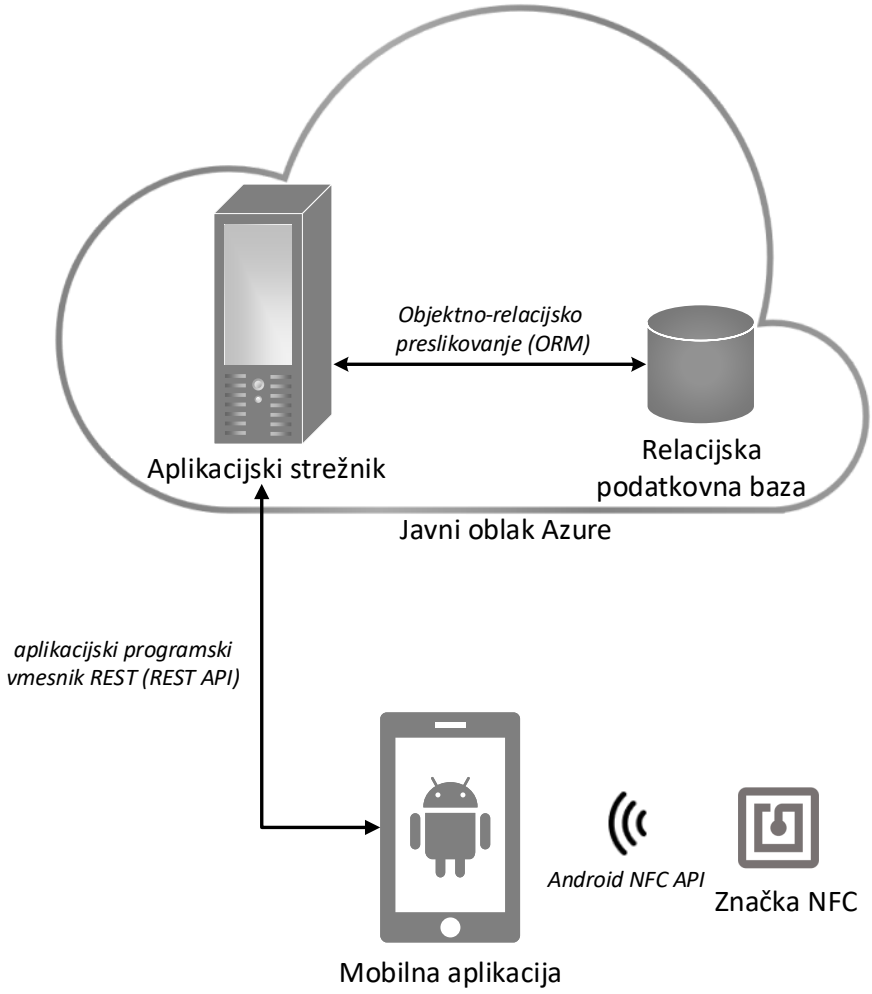
Zasnova in načrtovanje prototipa sistema

3.1 Ideja

Cilj magistrske naloge predstavlja zasnova, načrtovanje in izvedba sistema, ki podpira zaščito izdelkov, opremljenih z značko NFC, ter preverjanje avtentičnosti in sledljivosti izdelka v povezavi z oskrbovalno verigo. Predstavili bomo tri konkretne modele, ki bodo zagotavljali različne načine zaščite značk NFC, pri tem bo vsak naslednji model predstavljal višji nivo zaščite.

Prototip sistema sestavljajo zaledni del z aplikacijskim strežnikom in relacijsko podatkovno bazo ter mobilni del z aplikacijo na platformi Android (Slika 3.1). Mobilna aplikacija komunicira z značkami NFC preko programskega vmesnika NFC (angl. *Android NFC API*) in z aplikacijskim strežnikom preko aplikacijskega programskega vmesnika REST (angl. *REST API).* Aplikacijski strežnik dostopa tudi do relacijske podatkovne baze, in sicer z objektno-relacijskim preslikovanjem (angl. *object-relational mapping*, ORM). Vsa komunikacija med aplikacijo in strežnikom poteka preko varnih komunikacijskih kanalov.

Glavna razloga za izbiro operacijskega sistema Android sta njegova razširjenost pri pametnih mobilnih telefonih ter dobra podpora sistema za delo z



Slika 3.1: Prototip sistema za zaščito izdelkov in ugotavljanje ponaredkov

značkami NFC. Ker želimo, da bo mobilna aplikacija delovala povsod, kjer je na voljo internetna povezava, bomo aplikacijski strežnik in podatkovno bazo postavili na oblačno storitveno platformo Azure.

Iz prototipa sistema bomo izpeljali tri konkretne modele z različnimi stopnjami zaščite. Vsak izmed njih bo uporabljal drug tip značke NFC ter drug način zaščite podatkov, ki se nahajajo v pomnilniku značke. Tabela 3.1 predstavlja njihove ključne lastnosti z medsebojno primerjavo.

Najbolj enostaven način za zaščito (Model 1) bo deloval z značko NFC

	Model 1	**Model 2**	**Model 3**
Tip značke NFC	NTAG213	MIFARE Ultralight C	MIFARE DESFire EV1 2K
Izbran kriptografski algoritem	brez	112-bitni 3DES (2 ključa)	128-bitni AES
Oblika podatkov na znački	130 naključnih znakov v formatu NDEF	130 naključnih znakov v formatu NDEF	130 naključnih znakov v formatu DESFire Standard Data File
Dodatna zaščita podatkov pri prenosu	brez	zgoščena vrednost podatkov po algoritmu SHA-256	zgoščena vrednost podatkov po algoritmu SHA-256 vključno s soljo, 2048-bitno šifriranje RSA

Tabela 3.1: Značke NFC – značilnosti, ki so povezane z zaščito izdelkov

izvedbe NTAG213, ki je med vsemi značkami najbolj razširjena. Drugi način (Model 2) predstavlja nadgradnjo prvega in bo deloval z značko izvedbe MIFARE Ultralight C, ki za zaščito pomnilnika kartice ponuja kriptografski algoritem DES. Tretji način (Model 3) pa bo uporabljal trenutno najbolj varno obliko značke NFC – značko serije MIFARE DESFire, ki za zaščito svojega pomnilnika uporablja kriptografski algoritem AES. Izbrali smo izvedbo DESFire EV1 2K, ki z 2 kB pomnilnika povsem zadošča našim potrebam po prostoru za shranjevanje podatkov.

Vključili bomo tudi oskrbovalno verigo in pripadajoče deležnike – proizvajalce, logistična podjetja, skladišča ter prodajna mesta. Testni primer oskrbovalne verige in njenih deležnikov je prikazan na Sliki 3.2. Odločitev za lastno implementacijo sistema oskrbovalne verige in ne za uporabo obstoječega sistema (na primer GS1 GDSN) je predvsem v dodatni kompleksnosti in odvisnosti, ki jo s seboj prinese tak sistem. Pri tem smo presodili, da bo za potrebe prototipa zadostovala zgoraj predstavljena implementacija.

Aktivnosti v posamezni fazi oskrbovalne verige so:

- **Enroll Tag**: vpis nove značke v sistem oskrbovalne verige (proizvajalec)
- **Add SCM Record**: dodajanje novega zapisa gibanja značke v oskrbovalno verigo (proizvajalec, logistično podjetje, skladišče, prodajno mesto)
- **Authenticate**: preverjanje avtentičnosti značke (vsak uporabnik in vsi deležniki v oskrbovalni verigi, ko je izdelek na prodajnem mestu)

Slika 3.2: Oskrbovalna veriga in njeni deležniki

Mobilna aplikacija bo zasnovana tako, da bo imela tri načine delovanja – za običajne uporabnike, registrirane uporabnike ter registrirane uporabnike z administratorskimi pravicami. Vpis nove značke NFC v sistem oskrbovalne

verige bodo lahko izvedli le registrirani uporabniki z administratorskimi pravicami v prvi fazi verige. Dodajanje novega zapisa gibanja značke NFC v oskrbovalno verigo bo na voljo vsem registriranim uporabnikom, ki so povezani z značko NFC v izbrani oskrbovalni verigi. Preverjanje avtentičnosti značke NFC bo na voljo vsem uporabnikom v zadnji fazi oskrbovalne verige.

3.2 Aktivnosti sistema

Čeprav proizvajalec značk NXP Semiconductors za vsako značko zatrjuje unikatne identifikatorje [17, 18, 21] (v primeru da je UID dolžine 7 bajtov, to pomeni $2^{56} \approx 7.2 \times 10^{16}$ različnih identifikatorjev), ta ne zadošča za varno in enolično identifikacijo značk v nekem sistemu, saj že obstaja več primerov kloniranja značk, posledično kopiranja UID. Dodatna zaščita, s katero lahko onemogočimo kloniranje, je zapis vsebine na značko in omejitev dostopa do pomnilnika. Zapisana vsebina je nato skupaj z UID uporabljena za enolično identifikacijo značke v sistemu. Možno bi bilo zapisati podatke o izdelku, vendar je to preveč predvidljivo za Model 1, kjer podatki niso zaščiteni, zato smo se odločili za zapis naključnega niza. Način omejitve dostopa do pomnilnika značke NFC ter dolžina vsebine, ki jo lahko zapišemo na značko, sta odvisna od izbranega tipa značke in njenih načinov za izboljšavo varnosti.

Vpis nove značke NFC v sistem oskrbovalne verige

Prva aktivnost je vpis nove značke NFC v sistem oskrbovalne verige, kjer bomo vključili podatke o oskrbovalni verigi in proizvodu za shranjevanje na strežnik. V ozadju bomo zapisali tekstovno vsebino v pomnilnik značke NFC, ki bo kasneje koristila pri identifikaciji same značke. Vsebina, ki jo bomo zapisali v pomnilnik značke, bo naključno ustvarjena iz množice znakov x (3.1) in označena s simbolom s (3.2). Ker želimo modele med seboj neposredno primerjati, smo dolžino s pri vseh treh modelih omejili na 130 znakov. Razlog za to omejitev je v pomnilnikih značk NFC NTAG213 in Ultralight C, ki ne omogočata zapisa tekstovne vsebine v formatu NDEF,

daljše od 130 znakov.

$$x \in \{a-z, A-Z, 0-9, !, ?, ;, :, ., \&, \#, \$\} \tag{3.1}$$

$$s = RND_{130}(x) \tag{3.2}$$

Vrednost niza, UID značke, ključ za dostop do pomnilnika značke ter izbrano oskrbovalno verigo s podatki o proizvodu bomo posredovali na strežnik in na koncu shranili v podatkovno bazo. Zaščita podatkov je povezana z izbiro značke NFC in bo predstavljena s tremi postopki, ki so v nadaljevanju opisani kot Model 1, Model 2 in Model 3.

Dodajanje novega zapisa gibanja značke NFC v oskrbovalno verigo

Za zagotavljanje sledljivosti in preverjanje avtentičnosti značke NFC bomo po vsakem vpisu nove značke v sistem dodali tudi prvi zapis gibanja značke v oskrbovalno verigo – začetek oskrbovalne verige bo predstavljal prvi zapis proizvajalca. Nato bo vsak deležnik v oskrbovalni verigi ob prevzemu (angl. *receive*) ter odpremi (angl. *dispatch*) značke dodal po en zapis v oskrbovalno verigo. Zaporedje zapisov s časovnimi oznakami (angl. *timestamp*) tvori popolno sledljivost od proizvajalca do prodajnega mesta. Dodajanje zapisov gibanja značke NFC v oskrbovalno verigo bo omogočeno le tistim registriranim uporabnikom, ki so deležniki iste verige, kamor je vpisana značka. Aktivnost je v vseh fazah enaka za vse tri modele.

Preverjanje avtentičnosti značke NFC

Ko bo značko NFC prevzel zadnji deležnik v oskrbovalni verigi (običajno bo to prodajno mesto ali trgovina), bo lahko z mobilno aplikacijo izvedel preverjanje njene avtentičnosti in preveril podatke o sledljivosti proizvoda. Pri preverjanju se v ozadju najprej prebere vsebina pomnilnika značke in se nato pošlje na aplikacijski strežnik, kjer se izvede procesiranje. Pri tem

se upošteva podatke značke NFC in izdelka ter seznam zapisov gibanja v oskrbovalni verigi.

Postopek branja podatkov je za značke NFC brez zaščite pomnilnika (Model 1) povsem trivialen. Če ima značka svoj pomnilnik zaščiten pred nepooblaščenim dostopom (Model 2, Model 3), je treba pred branjem najprej znački poslati ustrezen ključ, da se v ozadju izvede protokol za avtentikacijo. Ključ za dostop je lahko enostaven (npr. 32-bitno geslo) ali bolj kompleksen (npr. 128-bitni šifrirni ključ pri algoritmu AES).

Preverjanje avtentičnosti se izvede kot notranji proces na aplikacijskem strežniku, kjer se preverijo spodnji pogoji, ki morajo v celoti veljati, da lahko potrdimo pristnost izdelka.

1. Enolična identifikacijska številka značke NFC (UID) mora biti v podatkovni bazi unikatna. To pomeni, da v celotnem sistemu ne obstajata dve znački, ki bi imeli enak UID, četudi je ena izmed njih označena kot prodana.

2. Zapisana vsebina v pomnilniku značke NFC se ujema z vsebino, shranjeno v podatkovni bazi:

   (a) Model 1 – primerja se vrednost niza, zapisanega v pomnilniku značke, z vrednostjo niza, shranjenega v podatkovni bazi.

   (b) Model 2 – primerja se zgoščena vrednost niza, zapisanega v pomnilniku značke, z zgoščeno vrednostjo niza, shranjenega v podatkovni bazi. Pri izračunu zgoščene vrednosti se uporabi zgoščevalni algoritem SHA-256.

   (c) Model 3 – primerja se zgoščena vrednost niza, zapisanega v pomnilniku značke, z zgoščeno vrednostjo niza, shranjenega v podatkovni bazi. Pri izračunu zgoščene vrednosti se uporabi zgoščevalni algoritem SHA-256 z dodano soljo. Mobilna aplikacija zgoščeno vrednost pri prenosu na aplikacijski strežnik dodatno šifrira s strežnikovim javnim ključem RSA. Strežnik dešifrira prejeto zgo-

ščeno vrednost niza s svojim zasebnim ključem RSA in nato izvede primerjavo.

3. Značko NFC je v sistem oskrbovalne verige vpisal uporabnik, ki je bil registriran pred vpisom značke v sistem in je imel ob vpisu zadostne in veljavne administratorske pravice.

4. Značka NFC je bila v sistem oskrbovalne verige dodana pred vsemi njenimi zapisi gibanja v oskrbovalni verigi.

5. Vsi zapisi gibanja značke NFC v sistemu oskrbovalne verige sledijo vnaprej predpisanemu vrstnemu redu deležnikov oskrbovalne verige ter so v pravilnem časovnem zaporedju.

6. Zadnji zapis gibanja značke NFC v sistemu oskrbovalne verige je prodajno mesto oziroma trgovina.

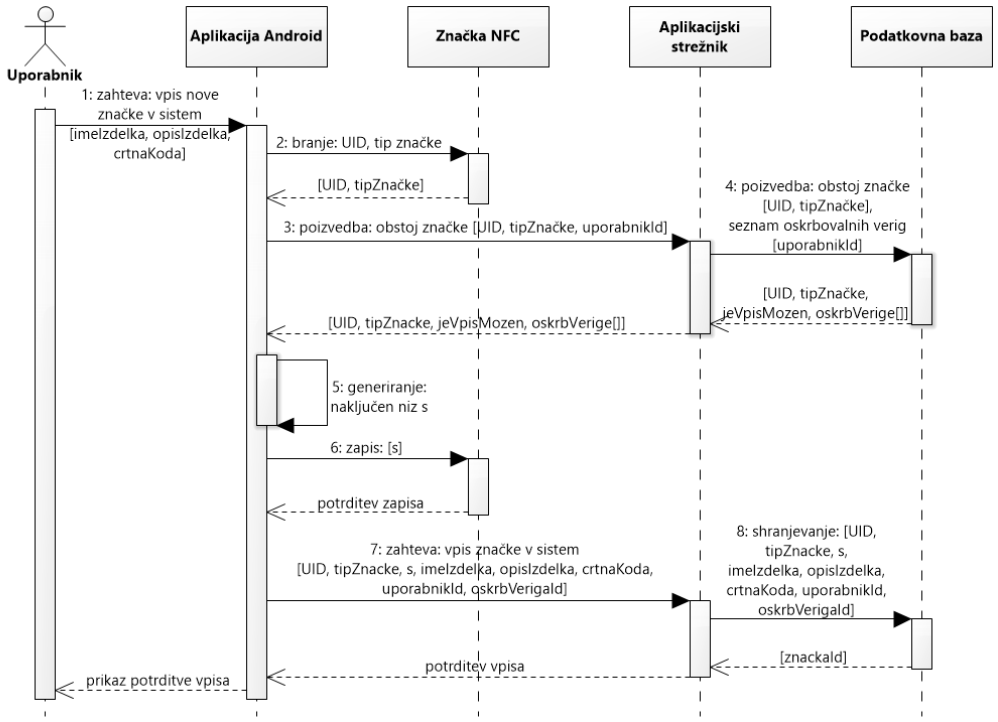
7. V času preverjanja avtentičnosti značka NFC v sistemu še ni bila označena kot prodana. Značko kot prodano označi prodajalec (kot zadnji deležnik oskrbovalne verige) v trenutku, ko potrošnik kupi izdelek.

3.3 Model 1

Najprej bomo v oskrbovalni verigi predstavili uporabo značke NFC (NTAG-213), ki ne uporablja zaščite pomnilnika. Ker ima na voljo le 144 B pomnilnika, namenjenega uporabniku, lahko vanj zapišemo do največ 130 znakov v obliki sporočila NDEF.

3.3.1 Vpis nove značke v sistem oskrbovalne verige

Na vsako novo značko NFC bomo pri vpisu v sistem oskrbovalne verige zapisali naključen niz s (3.2), ki bo kasneje uporabljen v postopku preverjanja avtentičnosti. Slika 3.3 prikazuje postopek vpisa, ki vključuje:



Slika 3.3: Model 1 – Vpis nove značke v sistem oskrbovalne verige

1. Uporabnik v mobilni aplikaciji odpre zavihek *Enroll tag*, vpiše ime, opis in črtno kodo izdelka, na katerega je pritrjena značka NFC, ji približa svoj mobilni telefon in sproži zahtevo za vpis v sistem.

2. Mobilna aplikacija prebere UID in tip značke NFC.

3. Mobilna aplikacija pošlje podatke na aplikacijski strežnik skupaj z uporabniškim imenom, kot poizvedbo po obstoju značke z enakim UID ter po seznamu oskrbovalnih verig, ki so uporabniku na voljo.

4. Aplikacijski strežnik v podatkovni bazi izvede preverjanje za morebiten obstoj značke ter pridobi seznam oskrbovalnih verig. V primeru, da značka z enakim UID še ni vpisana v sistem, strežnik odgovori mobilni aplikaciji s potrdilom za vpis nove značke v sistem in seznamom oskrbovalnih verig.

5. Mobilna aplikacija na podlagi prejetega potrdila ustvari naključen niz s (3.2) iz množice znakov x (3.1).

6. Niz s se zapiše v pomnilnik značke NFC.

7. Mobilna aplikacija pošlje zahtevo za vpis značke NFC na strežnik, skupaj z njenim UID, vrednostjo naključnega niza ter izbrano oskrbovalno verigo.

8. Strežnik shrani nov zapis o znački in o izdelku v podatkovno bazo ter odgovori mobilni aplikaciji s potrditvijo zapisa. Slednja nato obvestilo o uspešnem vpisu prikaže uporabniku na zaslonu.

3.3.2 Dodajanje novega zapisa gibanja značke v oskrbovalno verigo

Po vpisu značke v sistem oskrbovalne verige sledi dodajanje zapisov gibanja značke. Postopek, ki je enak v vseh fazah oskrbovalne verige, je prikazan na Sliki 3.4. Sestavljajo ga naslednji koraki:

1. Uporabnik v mobilni aplikaciji odpre zavihek *Add SCM Record* in približa svoj mobilni telefon znački NFC.

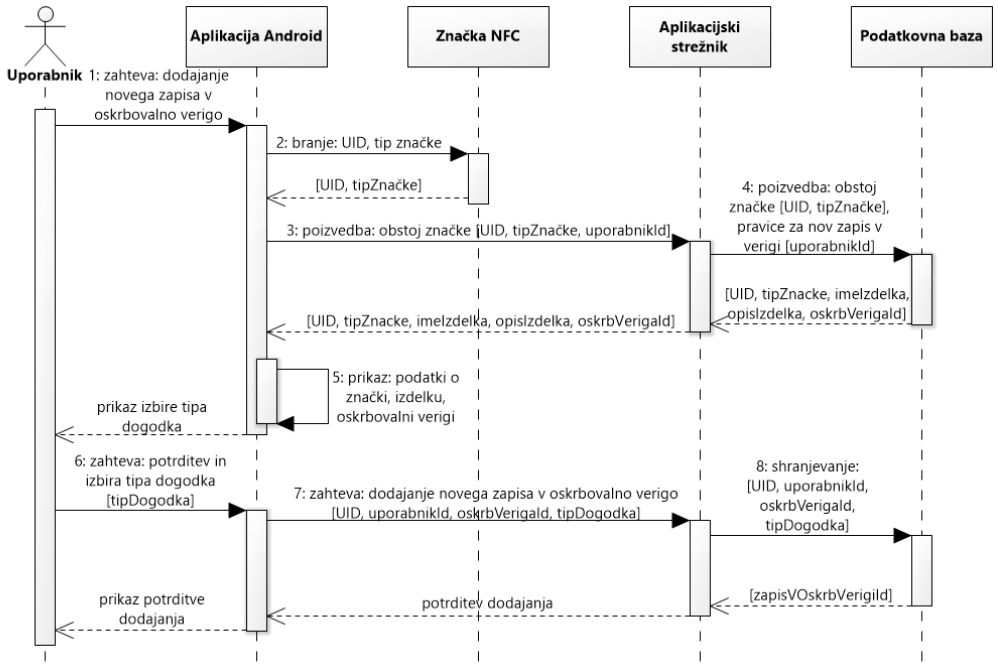
2. Mobilna aplikacija prebere UID in tip značke NFC.

3. Podatki se skupaj s poizvedbo po obstoju značke pošljejo na aplikacijski strežnik.

4. Strežnik izvede poizvedbo po podatkih izdelka in oskrbovalni verigi značke. Rezultat poizvedbe pošlje nazaj mobilni aplikaciji.

5. Ob prejemu potrdila ta prikaže podatke o izdelku (ime, opis in črtna koda izdelka) in oskrbovalni verigi na zaslonu.

6. Uporabnik preveri, ali so izpisani podatki pravilni, in izbere tip dogodka – prevzem ali odprema značke.



Slika 3.4: Model 1 – Dodajanje novega zapisa gibanja značke v oskrbovalno verigo

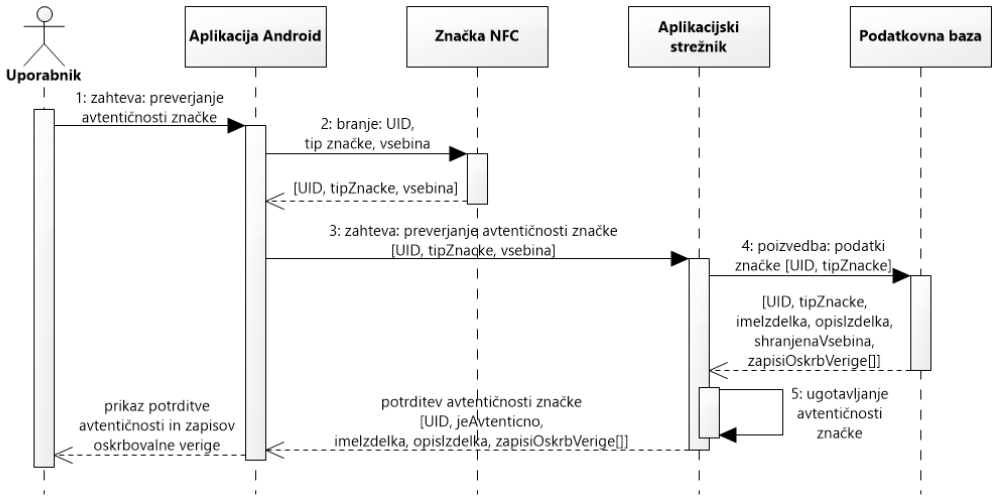
7. Mobilna aplikacija v naslednjem koraku pošlje na strežnik zahtevo po dodajanju novega zapisa gibanja značke, skupaj s pripadajočim tipom dogodka.

8. Strežnik ob prejemu zahteve shrani nov zapis gibanja v podatkovno bazo in pošlje mobilni aplikaciji potrditev dodajanja zapisa. Na koncu mobilna aplikacija izpiše potrditev na zaslonu.

3.3.3 Preverjanje avtentičnosti značke (NTAG213)

Vsak uporabnik mobilne aplikacije lahko pridobi informacijo o pristnosti izdelka s postopkom preverjanja avtentičnosti značke NFC, s katero je opremljen (Slika 3.5).

Postopek se izvede v zadnji fazi, kjer so na voljo vsi zapisi iz celotne



Slika 3.5: Model 1 – Preverjanje avtentičnosti značke

oskrbovalne verige, in vključuje:

1. Uporabnik z odpiranjem zavihka *Authenticate* v mobilni aplikaciji in s približanjem mobilnega telefona znački NFC sproži zahtevo za preverjanje.

2. Mobilna aplikacija prebere UID in tip značke NFC ter celotno vsebino njenega pomnilnika.

3. Po pridobitvi teh podatkov jih aplikacija pošlje na aplikacijski strežnik kot zahtevo za preverjanje avtentičnosti.

4. Aplikacijski strežnik ob sprejemu zahteve pridobi iz podatkovne baze vse podatke o izdelku in zapisih gibanja v oskrbovalni verigi.

5. Strežnik sproži notranji postopek ugotavljanja avtentičnosti na podlagi pridobljenih podatkov. V zadnjem koraku pošlje obvestilo o zavrnitvi, če avtentikacija ni bila uspešna, ali pa potrditev s podatki o izdelku in zapisih gibanja v oskrbovalni verigi mobilni aplikaciji. Slednja na koncu vse prejete podatke izpiše na zaslonu. Če uporabnik ni prijavljen, potem se podatki o vseh fazah v verigi ne izpišejo.

3.4 Model 2

Model 2 predstavlja izboljšano varianto zaščite podatkov in uporablja značko NFC izvedbe MIFARE Ultralight C, ki uporablja algoritem šifriranja 3DES (2 ključa). Tudi tukaj se pri vpisu značke v sistem zaradi omejitve uporabniškega pomnilnika vpiše naključen niz dolžine 130 znakov (3.2). Dodajanje novega zapisa gibanja značke v oskrbovalno verigo poteka na enak način kot v prejšnjem primeru za Model 1 (Poglavje 3.3.2).

3.4.1 Vpis nove značke v sistem oskrbovalne verige

Vpis nove značke NFC (MIFARE Ultralight C) v sistem lahko predstavimo v dveh delih (Slika 3.6). V korakih od 1 do 4 je postopek enak opisu za Model 1 (Poglavje 3.3.1), nato pa se izvedejo naslednji koraki:

5. Mobilna aplikacija na podlagi prejetega potrdila za vpis nove značke ustvari naključen niz s (3.2) iz množice znakov x (3.1) ter izračuna zgoščeno vrednost niza s po algoritmu SHA-256, označeno z y (3.3).

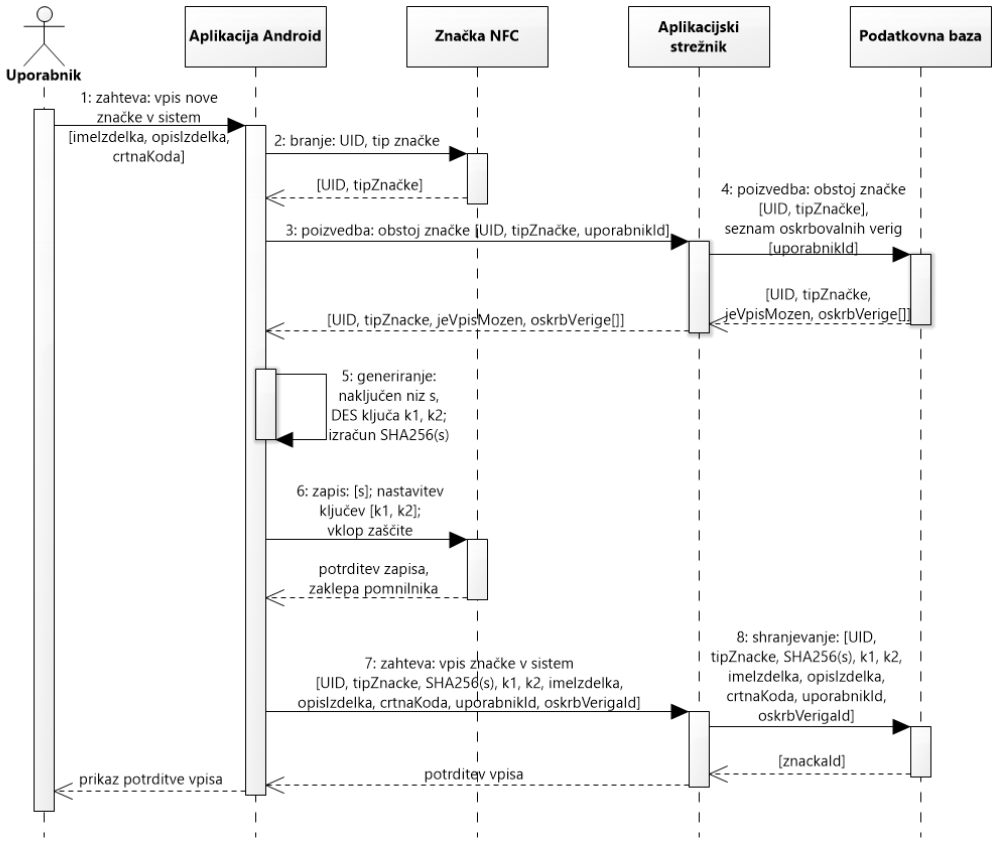
$$y = SHA_{256}(s) \tag{3.3}$$

   Aplikacija ustvari še dva 8-bajtna ključa DES k_1 in k_2 (3.4), vrednosti katerih sta naključno pridobljeni iz nabora znakov x (3.1).

$$k_1 = RND_{8B}(x), k_2 = RND_{8B}(x) \tag{3.4}$$

6. Nato mobilna aplikacija pošlje niz s za vpis v pomnilnik značke NFC, ji dodeli ključa DES k_1 in k_2 ter omogoči zaščito pred nepooblaščenim dostopom do pomnilnika.

7. Aplikacija pošlje na strežnik zahtevo za vpis nove značke v sistem oskrbovalne verige skupaj z UID in tipom značke, zgoščeno vrednostjo y, ključema DES k_1 in k_2 ter s podatki o izdelku in identifikatorjem izbrane oskrbovalne verige.



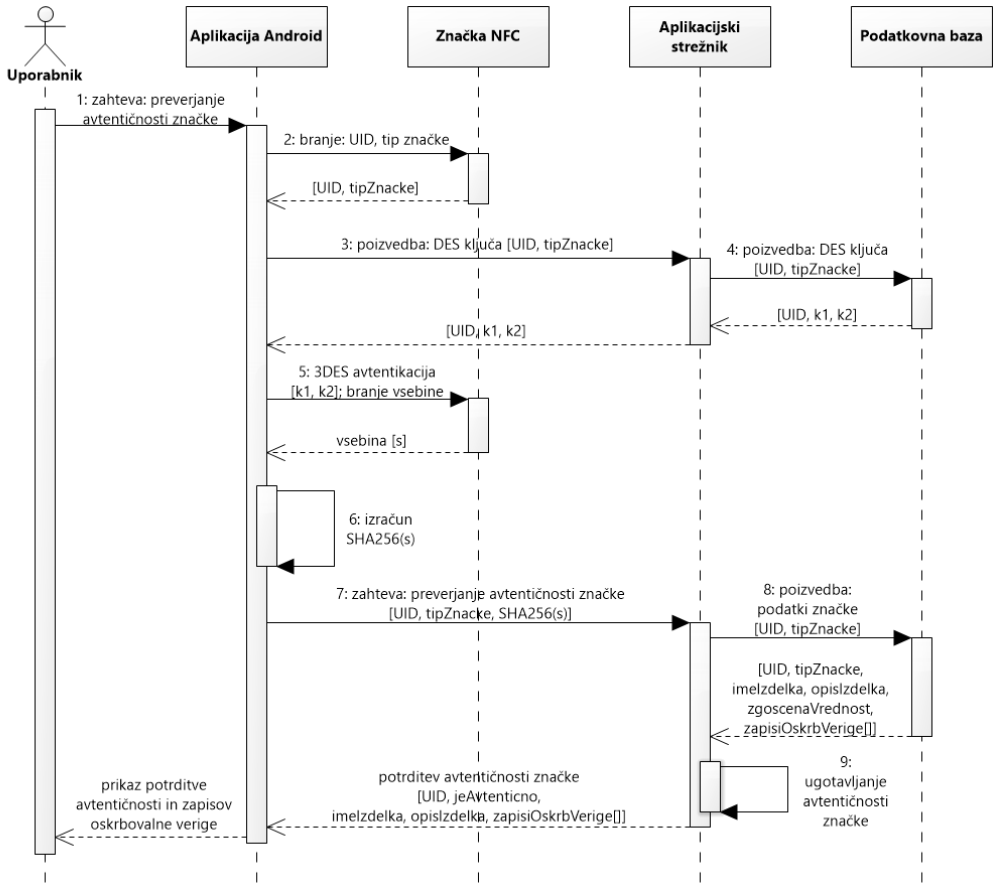
Slika 3.6: Model 2 – Vpis nove značke v sistem oskrbovalne verige

8. Strežnik shrani nov zapis o znački NFC v podatkovno bazo skupaj s pripadajočima ključema, podatkih o izdelku in oskrbovalni verigi. Nato odgovori mobilni aplikaciji z obvestilom o zaključku postopka, ki ga leta na koncu prikaže uporabniku na zaslonu.

3.4.2 Preverjanje avtentičnosti značke (MIFARE Ultralight C)

Aktivnost preverjanja avtentičnosti značke je do vključno drugega koraka enaka Modelu 1 (Slika 3.7), razlikovati se prične od tretjega koraka dalje:

3. Mobilna aplikacija pošlje na aplikacijski strežnik poizvedbo po ključih



Slika 3.7: Model 2 – Preverjanje avtentičnosti značke

DES, skupaj z UID in tipom značke.

4. Strežnik v podatkovni bazi izvede prejeto poizvedbo in dobljena ključa DES pošlje mobilni aplikaciji, skupaj z UID.

5. Aplikacija ob prejemu ključev izvede protokol za avtentikacijo 3DES (protokol je podrobneje opisan v *Protokol 3DES za avtentikacijo MIFARE Ultralight C*) in po njegovi uspešni izvedbi prebere vsebino pomnilnika značke.

6. Aplikacija izračuna še pripadajočo zgoščeno vrednost prebrane vsebine po algoritmu SHA-256.

7. Nato aplikacija pošlje na strežnik zahtevo po preverjanju avtentičnosti značke, ki vsebuje UID in tip značke ter zgoščeno vrednost, dobljeno v prejšnjem koraku.

8. Strežnik ob prejemu zahteve izvede poizvedbo po shranjeni zgoščeni vrednosti, podatkih o izdelku ter po seznamu zapisov gibanja značke v oskrbovalni verigi.

9. Nad prejetimi podatki se izvede postopek za ugotavljanje avtentičnosti. Rezultat preverjanja se obravnava enako kot za Model 1 (Poglavje 3.3.3, korak 5).

Protokol 3DES za avtentikacijo MIFARE Ultralight C

Protokol 3DES za avtentikacijo med bralnikom NFC in pametno kartico MIFARE Ultralight C je opisan v tehnični dokumentaciji [17]. Na Sliki 3.8 je prikazan postopek:

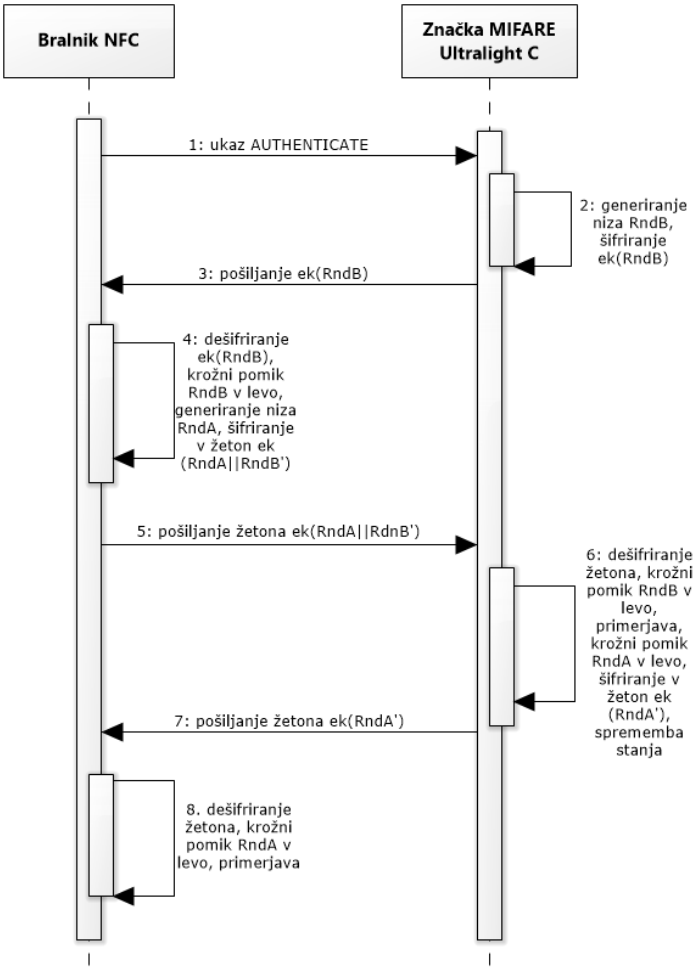
1. Protokol za avtentikacijo vedno sproži bralnik NFC s pošiljanjem ukaza *Authenticate* znački MIFARE Ultralight C, ki se nahaja v njegovem obsegu delovanja.

2. Značka ob prejemu ukaza generira 8-bajtni naključni niz $RndB$ in ga šifrira s ključem k (oba, bralnik in značka, poznata ključ).

3. Šifriran niz nato pošlje nazaj bralniku, kar označimo kot $ek(RndB)$.

4. Bralnik NFC nato dešifrira prejet niz $ek(RndB)$ s ključem k in s tem pridobi niz $RndB$. V naslednjem koraku ga krožno pomakne za 8 bitov v levo – operacija krožnega pomika v levo (angl. *circular left shift*) in tako nastane $RndB'$. Bralnik generira še 8-bajtni naključni niz $RndA$, zraven pripne (angl. *concatinate*) niz $RndB'$ in dobljeno zašifrira s ključem k, kar označimo z $ek(RndA||RndB')$.

5. Šifrirana vrednost predstavlja žeton, ki ga bralnik pošlje znački.



Slika 3.8: Protokol 3DES za avtentikacijo značke MIFARE Ultralight C

6. Značka ob prejemu žetona izvede dešifriranje in pridobi vrednosti nizov $RndA$ in $RndB'$. Niz $RndB$, poslan v koraku (3), značka krožno pomakne za 8 bitov v levo in dobljeno vrednost primerja z vrednostjo $RndB'$. Če se vrednosti ne ujemata, značka pošlje bralniku sporočilo o napaki in s tem zaključi proces avtentikacije. V primeru, da sta vrednosti enaki, značka izvede še zadnji korak – krožno pomakne niz $RndA$ za 8 bitov v levo. Nastane niz $RndA'$, ki ga nato značka šifrira s ključem k. Šifrirana vrednost predstavlja žeton.

7. Značka v naslednjem koraku pošlje žeton bralniku, svoje stanje nastavi kot *authenticated* in s tem dovoli bralniku dostop do svojega pomnilnika.

8. Bralnik dešifrira prejet žeton s ključem k in dobljeno vrednost primerja z vrednostjo poslanega niza $RndA$, ki ga pred tem še krožno pomakne za 8 bitov v levo. Če se vrednosti ujemata, bralnik s tem pridobi informacijo, da sta oba uporabila enak ključ DES med protokolom avtentikacije in da mu je značka omogočila dostop do njenega pomnilnika. V primeru, da sta vrednosti različni, bralnik zaključi proces avtentikacije in prekine povezavo z značko.

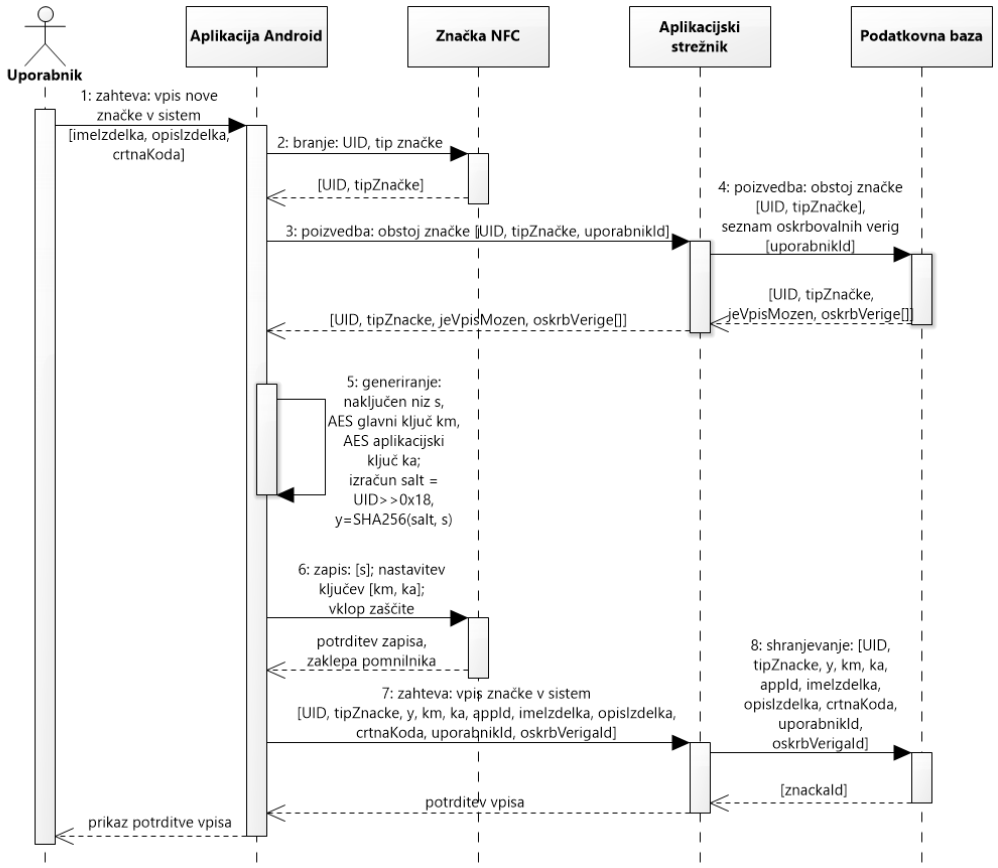
3.5 Model 3

Model 3 predstavlja rešitev sistema z najvišjo stopnjo varnosti in uporablja značko NFC izvedbe MIFARE DESFire EV1, ki ima zaščito pomnilnika z algoritmom za šifriranje AES. Dodajanje novega zapisa gibanja značke v oskrbovalno verigo je enako za vse tipe značk in je bilo opisano za Model 1 (Poglavje 3.3.2), vpis nove značke in preverjanje avtentičnosti pa sta predstavljeni v nadaljevanju. Preverjanje avtentičnosti značke je nadgrajeno z vključitvijo šifriranja podatkov (RSA) pri prenosu vsebine med mobilno aplikacijo in aplikacijskim strežnikom.

3.5.1 Vpis nove značke v sistem oskrbovalne verige

Vpis nove značke NFC (MIFARE DESFire EV1) v sistem lahko predstavimo v dveh delih (Slika 3.9). V korakih od 1 do 4 je postopek enak opisu za Model 1 (Poglavje 3.3.1), nato pa se izvedejo naslednji koraki:

5. Mobilna aplikacija ustvari naključen niz s (3.2) iz množice znakov x (3.1) in zanj izračuna zgoščeno vrednost y (3.5) po algoritmu SHA-256, kateremu dodamo tudi sol (angl. *salt*) in s tem preprečimo napad z mavrično tabelo. Sol *salt* (3.6) izračuna tako, da UID značke pomakne



Slika 3.9: Model 3 – Vpis nove značke v sistem oskrbovalne verige

za 24 bitov v desno (0x18 v šestnajstiškem zapisu) in dobi 4-bajtno vrednost, ki je dovolj različna za značke iz iste serije in jo je smiselno uporabiti za sol.

$$y = SHA_{256}(salt, s) \tag{3.5}$$

$$salt = UID >> 0\text{x}18 \tag{3.6}$$

Nato aplikacija generira še dva 128-bitna ključa AES (3.7), vrednosti katerih sta naključno pridobljeni iz nabora znakov x (3.1). Prvi ključ je glavni ključ k_m (angl. *master key*), drugi pa aplikacijski ključ k_a (angl. *application key*). Oba ključa bo mobilna aplikacija kasneje

potrebovala za izvedbo protokola AES za avtentikacijo.

$$k_m = RND_{128bit}(x), k_a = RND_{128bit}(x) \tag{3.7}$$

6. Mobilna aplikacija nastavi znački glavni ključ k_m, formatira njen pomnilnik in v njenem pomnilniku ustvari novo aplikacijo. V to aplikacijo nato vstavi tekstovno datoteko, ki vsebuje naključen niz s, ter na koncu aplikaciji dodeli še aplikacijski ključ k_a.

7. Po končanem zapisovanju mobilna aplikacija pošlje na strežnik zahtevo za vpis nove značke v sistem oskrbovalne verige. Zahteva vsebuje UID in tip značke, zgoščeno vrednost y, ključa AES k_m in k_a, identifikator aplikacije ter podatke o izdelku in izbrani oskrbovalni verigi.

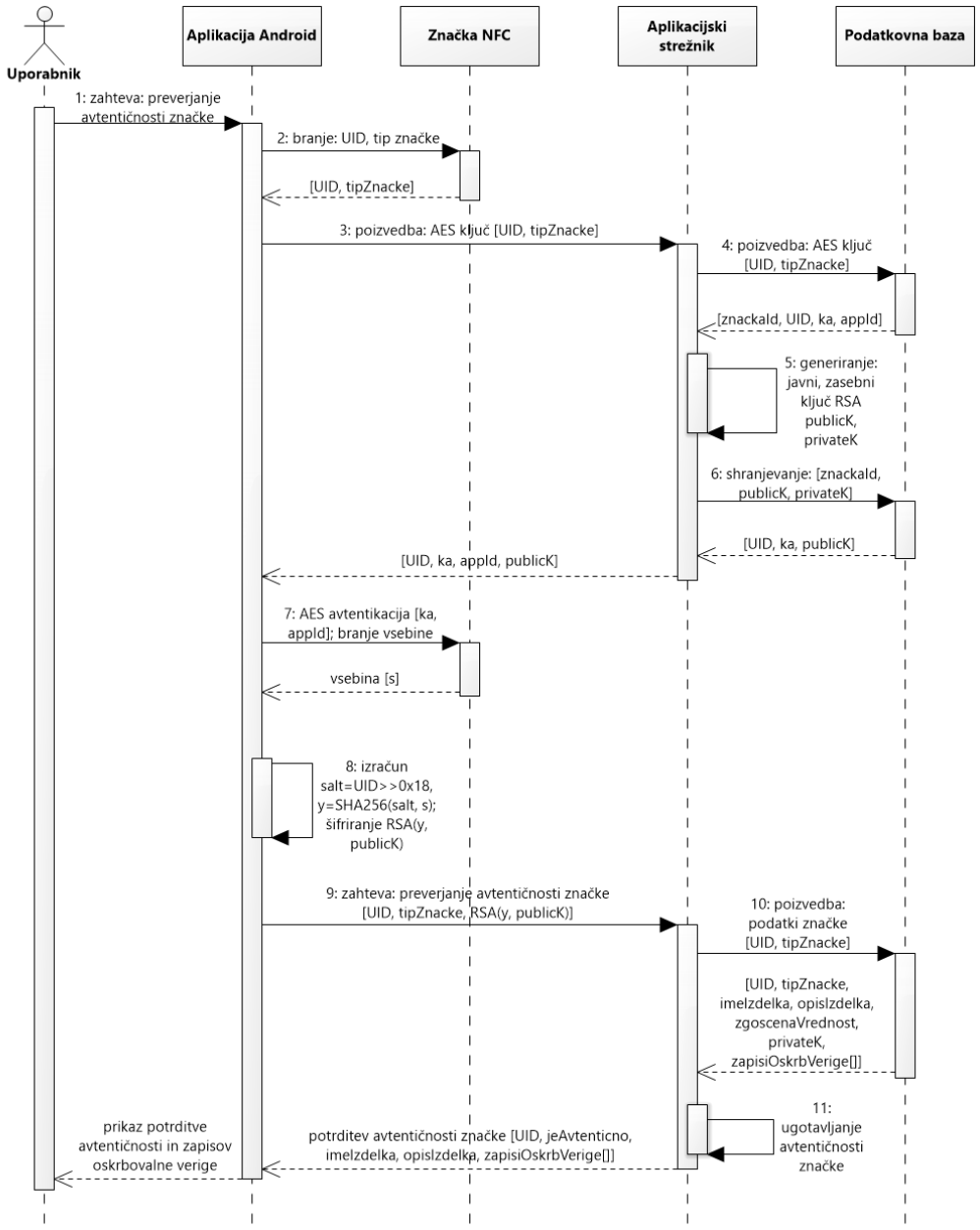
8. Strežnik ob prejemu zahteve shrani vse pripadajoče podatke v podatkovno bazo in mobilni aplikaciji vrne potrdilo o vpisu značke v sistem oskrbovalne verige, ki ga na koncu uporabnik vidi izpisanega na zaslonu.

3.5.2 Preverjanje avtentičnosti značke (MIFARE DESFire EV1)

Postopek preverjanja avtentičnosti (Slika 3.10) je na začetku enak kot za Model 1 in Model 2. Razlikuje se od koraka 3 naprej in vključuje:

3. Mobilna aplikacija pošlje na strežnik zahtevo za aplikacijski ključ AES, skupaj z UID in tipom značke. Glavni ključ značke ostane na strežniku in se ne pošilja, saj z njim ni mogoče pridobiti dostopa za branje vsebine pomnilnika značke. Namenjen je le za upravljanje s pomnilnikom značke.

4. Strežnik v podatkovni bazi izvede poizvedbo po značkinem aplikacijskem ključu AES.



Slika 3.10: Model 3 – Preverjanje avtentičnosti značke

5. Po končani poizvedbi strežnik ustvari še nov par javnega in zasebnega ključa RSA.

6. Oba ključa nato zapiše v podatkovno bazo za kasnejšo uporabo pri dešifriranju in mobilni aplikaciji vrne aplikacijski ključ AES, identifikator pripadajoče aplikacije ter svoj javni ključ RSA.

7. Mobilna aplikacija po prejemu odgovora strežnika uporabi aplikacijski ključ AES in identifikator aplikacije za izvedbo protokola AES za avtentikacijo (*Protokol AES za avtentikacijo MIFARE DESFire EV1*). Po uspešno izvedenem protokolu uporabi sejni ključ AES (angl. *session key*), s katerim pridobi dostop do pomnilnika značke in prebere vsebino datoteke.

8. Aplikacija izračuna še zgoščeno vrednost prebrane vsebine po algoritmu SHA-256, ki mu doda sol (to izračuna na enak način kot pri aktivnosti vpisa nove značke v sistem oskrbovalne verige (3.6)). Zgoščeno vrednost šifrira še s strežnikovim javnim ključem RSA.

9. Mobilna aplikacija zgoščeno vrednost skupaj z UID in tipom značke pošlje na strežnik kot zahtevo za preverjanje avtentičnosti značke.

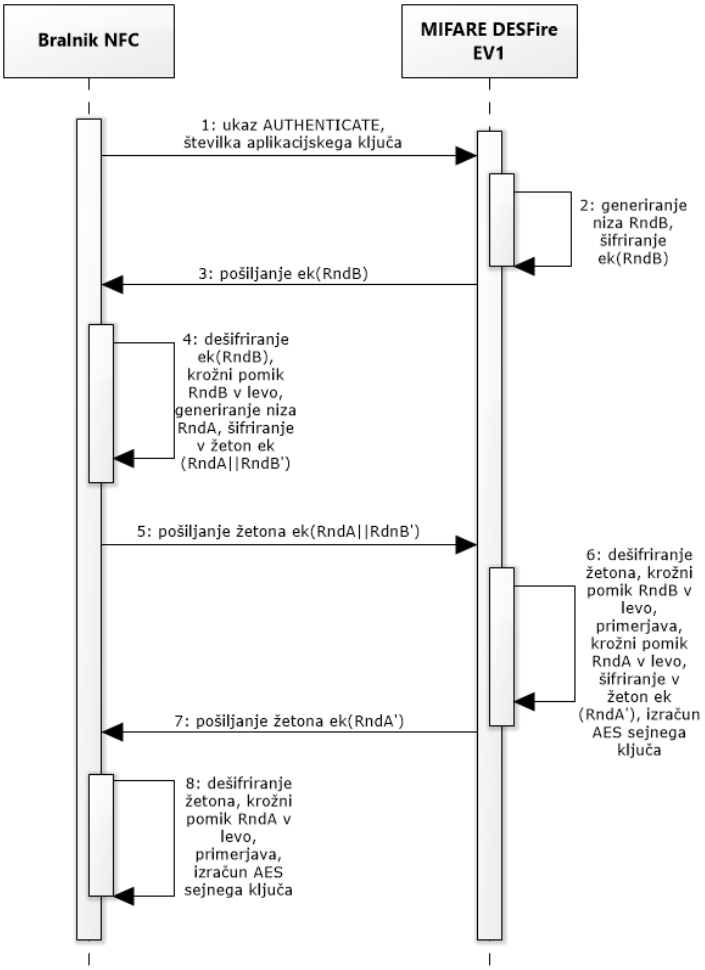
10. Strežnik izvede poizvedbo po podatkih o izdelku, zgoščeni vrednosti in zasebnem ključu RSA ter po seznamu gibanja značke v oskrbovalni verigi.

11. Strežnik izvede še postopek za ugotavljanje avtentičnosti značke na podlagi pridobljenih podatkov. Rezultat preverjanja se obravnava enako kot za Model 1 (Poglavje 3.3.3, korak 5).

Protokol AES za avtentikacijo MIFARE DESFire EV1

Avtentikacija med bralnikom NFC in značko MIFARE DESFire EV1 je natančneje opisana v tehnični dokumentaciji izdelka [18, 40]. Slika 3.11 prikazuje njegovo izvedbo v naslednjih korakih:

1. Protokol za avtentikacijo vedno sproži bralnik NFC s pošiljanjem ukaza *Authenticate* ter številko aplikacijskega ključa AES znački MIFARE DESFire EV1, ki se nahaja v njegovem dosegu delovanja.



Slika 3.11: Protokol AES za avtentikacijo značke MIFARE DESFire EV1

2. Značka ob prejemu ukaza generira 16-bajtni naključni niz $RndB$ in ga šifrira z ustreznim aplikacijskim ključem AES k.

3. Šifriran niz nato pošlje nazaj bralniku, kar označimo kot $ek(RndB)$.

4. Bralnik NFC dešifrira prejet niz s ključem k in s tem pridobi vrednost niza $RndB$. V naslednjem koraku s krožnim pomikom za 8 bitov v levo izračuna še niz $RndB'$. Generira še 16-bajtni naključni niz $RndA$, mu poleg pripne niz $RndB'$ in dobljeno šifrira s ključem k, kar označimo z

$ek(RndA||RndB')$ in predstavlja vrednost žetona.

5. Pridobljen žeton nato bralnik pošlje znački.

6. Značka od prejemu žetona izvede njegovo dešifriranje. S tem pridobi vrednosti nizov $RndA$ in $RndB'$. Niz $RndB$, poslan v koraku (3), značka krožno pomakne za 8 bitov v levo in dobljeno vrednost primerja z vrednostjo $RndB'$. Če se vrednosti ne ujemata, značka pošlje bralniku sporočilo o napaki in s tem zaključi proces avtentikacije. V primeru, da sta vrednosti enaki, značka izvede še zadnji korak – krožno pomakne niz $RndA$ za 8 bitov v levo, dobi niz $RndA'$ in ga nato šifrira s ključem k. Šifriran niz $ek(RndA')$ predstavlja novo vrednost žetona.

7. Žeton značka pošlje bralniku, sama pa izračuna sejni ključ $AES_{sessionKey}$ (3.8), s katerim je bralniku omogočila dostop do aplikacije.

8. Bralnik dešifrira dobljen žeton s ključem k in nato dobljeno vrednost primerja z vrednostjo poslanega niza $RndA$, ki ga pred tem še krožno pomakne za 8 bitov v levo. Če se vrednosti ujemata, bralnik s tem pridobi informacijo, da sta oba uporabila enak aplikacijski ključ AES in da mu je značka odobrila dostop do želene aplikacije. Izračuna še vrednost sejnega ključa AES, ki ga nato uporabi za dostop do aplikacije.

Vrednost sejnega ključa AES oba, bralnik NFC in značka, izračunata na enak način (3.8).

$$AES_{sessionKey} = RndA_{[0-3\ byte]} + RndB_{[0-3\ byte]} + \\ + RndA_{[12-15\ byte]} + RndB_{[12-15\ byte]} \tag{3.8}$$

Poglavje 4

Razvoj sistema

4.1 Mobilna aplikacija

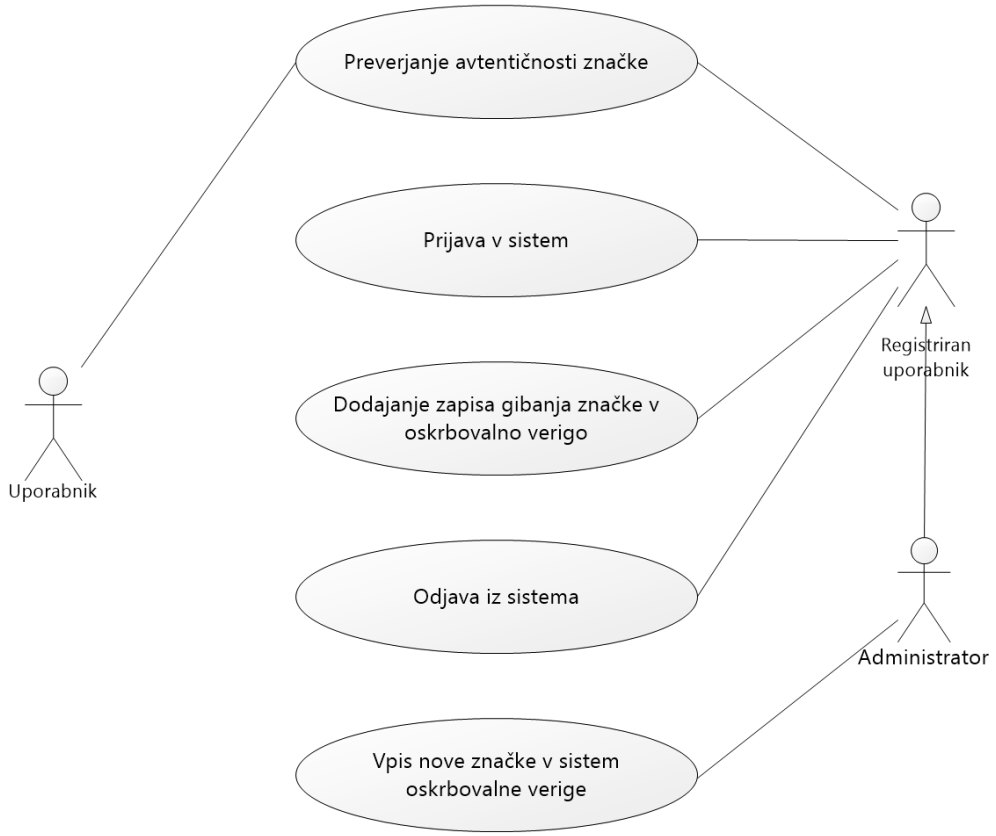
Mobilno aplikacijo smo razvili za operacijski sistem Android, ki je s 87,7 odstotka tržnega deleža vodilni pri prodaji novih pametnih telefonov [41]. S svojimi programskimi vmesniki API ponuja dobro povezljivost pri delu z značkami NFC. Zaradi preprostosti smo zasnovali eno aplikacijo, ki omogoča vse tri aktivnosti – vpis nove značke NFC v sistem, dodajanje novega zapisa gibanja značke ter preverjanje avtentičnosti značke. V realnem okolju bi bilo smiselno glede na aktivnosti v oskrbovalni verigi razviti tri ločene aplikacije, ki bi bile namenjene posamezni skupini uporabnikov.

Mobilno aplikacijo smo poimenovali *AuthentiTap* in predstavlja skovanko dveh angleških glagolov za avtentikacijo (angl. *authenticate*) in za hiter dotik zaslona (angl. *tap*). Aplikacija zahteva dovoljenji za uporabo adapterja NFC in interneta ter jo je možno naložiti na mobilne telefone, ki uporabljajo verzijo Android 5.0 ali novejše. Funkcionalnosti mobilne aplikacije so prilagojene tipu uporabnika. Za ta namen smo definirali dva tipa – običajen in registriran uporabnik.

Uporabnik ali običajen uporabnik je vsak uporabnik mobilne aplikacije, ki ni prijavljen v aplikacijo in nima računa za upravljanje z značkami

NFC in oskrbovalnimi verigami. V aplikaciji mu je omogočena le aktivnost preverjanja avtentičnosti značke. Tipičen predstavnik je potrošnik, ki v trgovini z mobilno aplikacijo preveri avtentičnost izdelka.

Registriran uporabnik ali administrator je uporabnik mobilne aplikacije, ki je prijavljen v aplikacijo s svojim računom za upravljanje z značkami NFC in oskrbovalnimi verigami. Če ima dodeljene administratorske pravice lahko dodaja nove značke NFC v sistem oskrbovalne verige. Na Sliki 4.1 so prikazane aktivnosti, ki jih lahko izvedejo uporabnik, registriran uporabnik in administrator.



Slika 4.1: Uporabniki in njihove aktivnosti v mobilni aplikaciji

Vpis nove značke v sistem oskrbovalne verige omogoča registriranemu uporabniku z administratorskimi pravicami vpis nove značke NFC v sistem oskrbovalne verige.

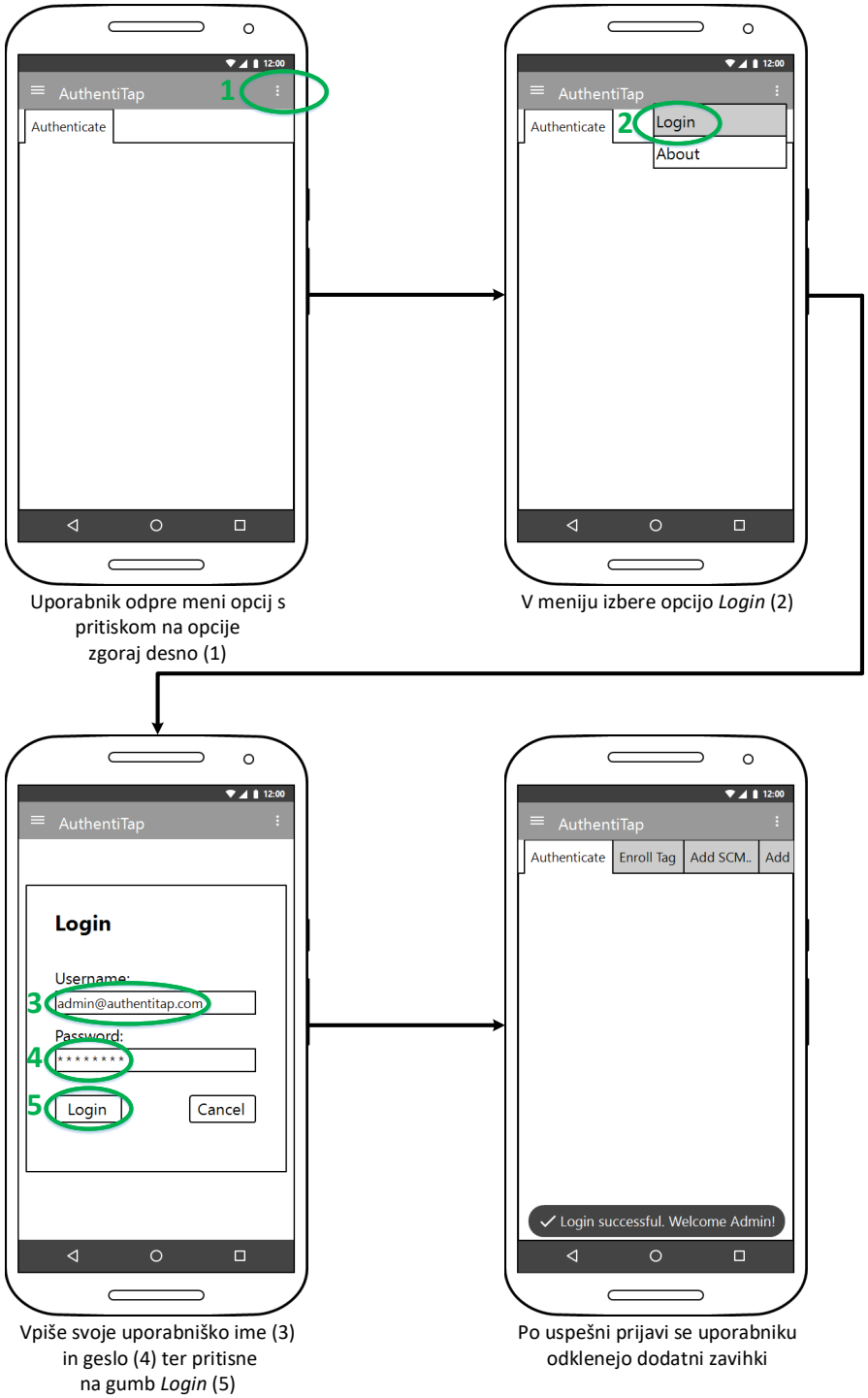
Preverjanje avtentičnosti značke je funkcionalnost, ki jo lahko izvedejo vsi tipi uporabnikov. Avtentičnost izdelka preverijo tako, da v prvem zavihku aplikacije poimenovanem *Authenticate* približajo svoj mobilni telefon k znački NFC.

Prijava v sistem je na voljo registriranemu uporabniku z obstoječim uporabniškim računom. V obrazcu za prijavo uporabnik vnese svoje uporabniško ime in geslo, se z njima prijavi v sistem in s tem pridobi dostop do ostalih funkcionalnosti.

Dodajanje zapisa gibanja značke v oskrbovalno verigo izvede registriran uporabnik ob dogodku odpreme ali prevzema izdelka.

Odjava iz sistema je na voljo registriranemu uporabniku (če se je pred tem uspešno prijavil v sistem) in mu omogoča izpis iz sistema in prehod aplikacije v stanje za običajnega uporabnika.

Skeletni model za prijavo uporabnika v sistem je prikazan na Sliki 4.2. Registriran uporabnik se v sistem prijavi tako, da v aplikaciji iz menija opcij zgoraj desno (1) izbere opcijo *Login* (2) in nato vpiše svoje uporabniško ime (3) ter geslo (4). S pritiskom na gumb *Login* (5) pošlje prijavne podatke na aplikacijski strežnik. V primeru, da je vnesel pravilne in veljavne podatke, uporabnik dobi na zaslonu izpisano obvestilo o uspešni prijavi, vidni mu postanejo še dodatni zavihki. V primeru, da uporabnik ne obstaja ali nima pravic za prijavo v sistem, se uporabniku na zaslonu izpiše napaka pri prijavi.

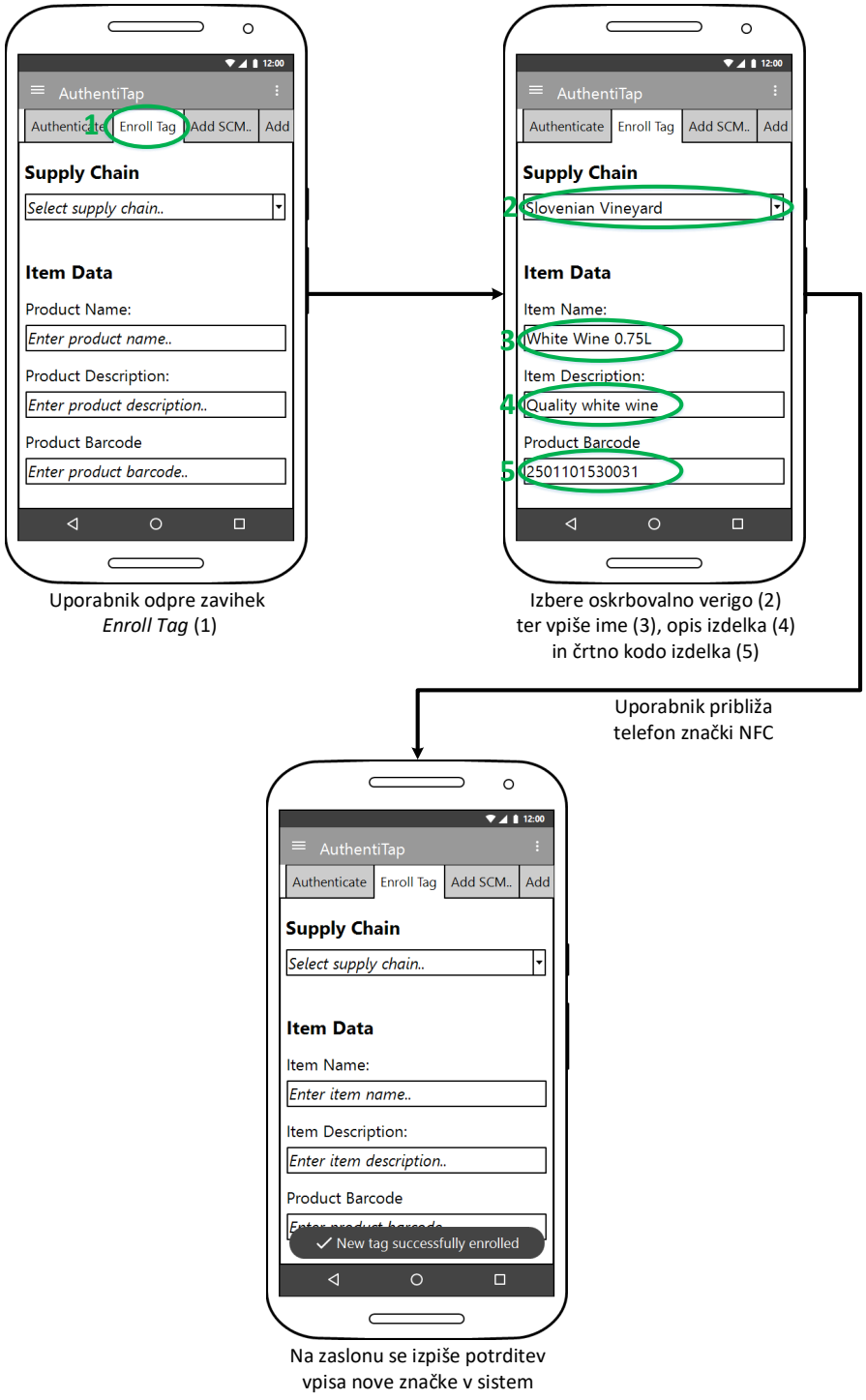


Slika 4.2: Prijava uporabnika

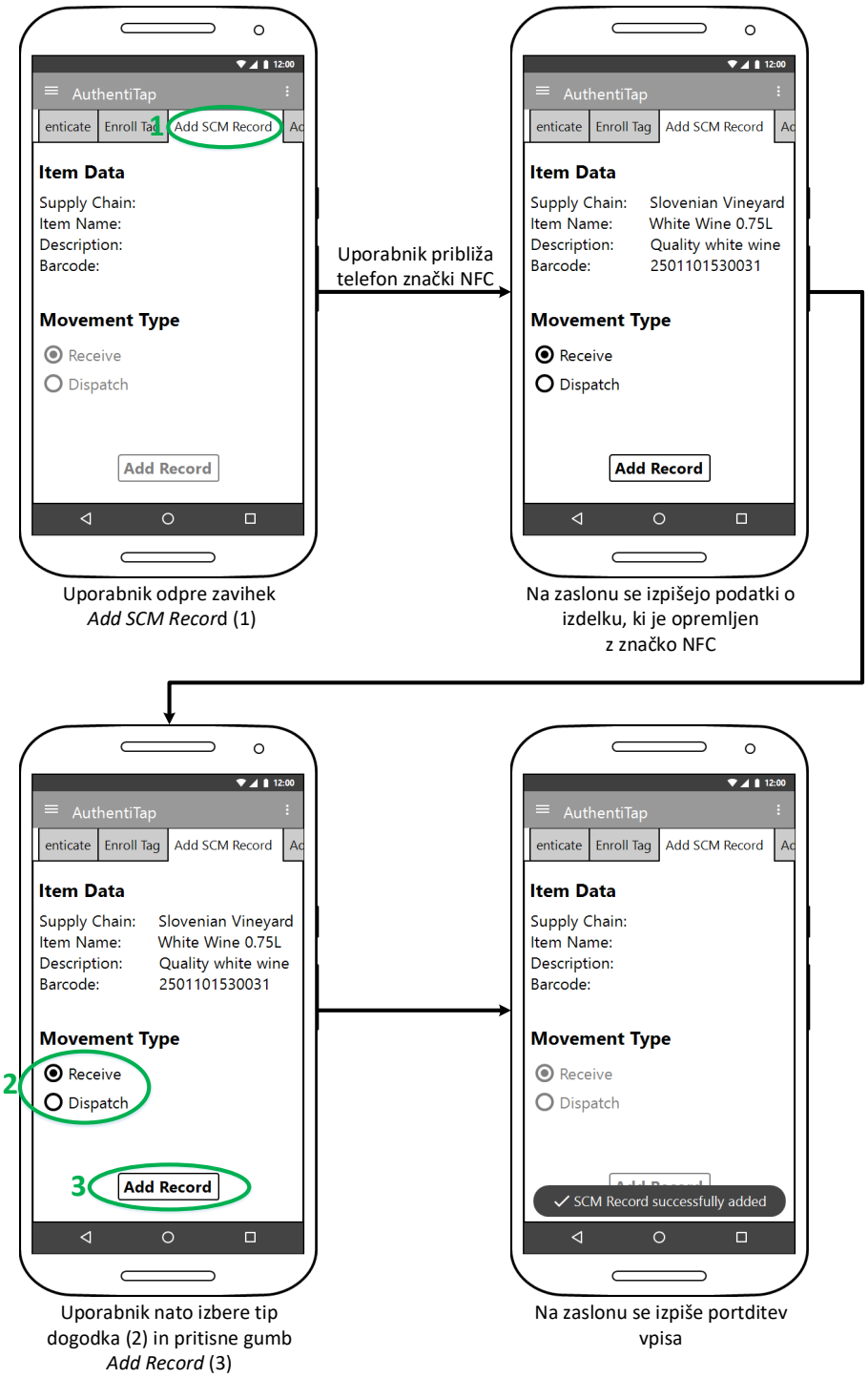
Prijavljen uporabnik, ki ima administratorske pravice, za vpis nove značke v sistem odpre zavihek *Enroll Tag* (1) in nato iz spustnega menija izbere ustrezno oskrbovalno verigo (2), kamor želi vpisati značko (Slika 4.3). Vpiše še ime (3), opis (4) in črtno kodo izdelka (5) ter približa telefon in na znački sproži vpis v sistem. Na koncu se uporabniku na zaslonu izpiše potrditev vpisa oziroma obvestilo o napaki pri vpisu, če ta ni bi uspešen.

Prijavljen uporabnik doda nov zapis gibanja značke v oskrbovalno verigo tako, da odpre zavihek *Add SCM Record* (1), kar v prevodu pomeni dodaj zapis gibanja v oskrbovalno verigo, in nato približa telefon znački NFC (Slika 4.4). Na zaslonu se mu izpišejo podatki o izdelku, ki je opremljen z značko. V naslednjem koraku uporabnik izbere tip dogodka gibanja (2) – dogodek prevzema (angl. *receive*) ali dogodek odpreme izdelka (angl. *dispatch*) in s pritiskom na gumb *Add Record* (3) pošlje nov zapis na strežnik. Ta ga ob prejemu shrani v podatkovno bazo in odgovori aplikaciji s potrditvijo vpisa novega dogodka.

Na Sliki 4.5 zgornja dva zaslona prikazujeta dogajanje pri preverjanju avtentičnosti značke, ki ga izvede običajen uporabnik, spodnja zaslona pa dogajanje, ki ga sproži registriran uporabnik. V mobilni aplikaciji odpre zavihek *Authenticate* (1) in nato približa svoj telefon znački. V ozadju se izvede aktivnost preverjanja avtentičnosti, njen rezultat se izpiše uporabniku na zaslonu. V primeru, da je preverjanje avtentičnosti izvedel registriran uporabnik, mu aplikacija na zaslonu prikaže tudi vse zapise gibanja značke NFC v sistemu oskrbovalne verige.



Slika 4.3: Vpis nove značke v sistem oskrbovalne verige



Slika 4.4: Dodajanje novega zapisa gibanja značke

Slika 4.5: Preverjanje avtentičnosti značke

4.2 Aplikacijski strežnik in podatkovna baza

Prototip sistema vključuje tudi aplikacijski strežnik in relacijsko podatkovno bazo, kjer se bo izvajala glavna aplikacijska logika. Strežnik bo ponujal programski vmesnik REST API, preko katerega bo potekala vsa komunikacija z mobilno aplikacijo. Programski vmesnik bo zaščiten navzven – za dostop do virov se bo moral vsak odjemalec (v našem primeru mobilna aplikacija) prijaviti preko protokola OAuth 2.0. Vsi podatki o značkah NFC (podatki o izdelku, pripadajoči ključi DES oziroma AES, zgoščene vrednosti zapisane vsebine) in o oskrbovalnih verigah bodo shranjeni v podatkovni bazi, do katere bo imel direkten dostop le aplikacijski strežnik. Vsa komunikacija med mobilno aplikacijo in aplikacijskim strežnikom bo potekala po varnem protokolu HTTPS/SSL, ki omogoča šifriran promet. Aplikacijski strežnik in podatkovno bazo bomo postavili v gostovanje na oblak Azure, da bosta vedno na voljo mobilni aplikaciji. Izdelali smo tudi spletno aplikacijo, ki predstavlja nadzorno ploščo, preko katere administrator dodaja nove uporabnike in oskrbovalne verige. Administratorju je omogočeno tudi dodeljevanje uporabnikov v verige, aktivacija in deaktivacija uporabnikov ter dodelitev ali odvzem njihovih administratorskih pravic.

4.3 Razvojno okolje

4.3.1 Android Studio

Android Studio je integrirano razvojno okolje (angl. *integrated development environment*, krajše IDE) za razvoj aplikacij na platformi Android, razvito s strani podjetja Google [42]. Orodje temelji na razvojnem okolju IntelliJ IDEA in deluje na vseh glavnih operacijskih sistemih Windows, macOS in Linux. Za razvijanje programske logike orodje uporablja programski jezik Java in označevalni jezik XML za delo z grafičnim vmesnikom. Android Studio za razvoj programske opreme na platformi Android uporablja paket Android SDK, ki vsebuje številne knjižnice, izseke kode, orodje za razhroščevanje,

dokumentacijo ter številne emulatorje oziroma posnemovalnike (angl. *emulators*) mobilnih telefonov, tablic in pametnih ur. Emulatorji razvijalcem omogočajo, da napisano kodo izvajajo na virtualnih napravah, ki posnemajo prave – s tem odpade potreba po fizičnih napravah. V Android Studiu je na voljo orodje za gradnjo projekta Gradle, ki iz izvornih datotek in vključenih knjižnic samodejno ustvari paket APK za distribucijo in namestitev aplikacije. Razvijalec lahko z urejanjem skripte gradle, ki jo uporablja orodje Gradle, spremeni potek gradnje projekta ali pa vključi dodatne knjižnice, ki jih potrebuje med razvojem. Pri razvoju smo uporabili Android Studio različice 2.3.3, od oktobra 2017 pa je na voljo že nova različica 3.0. Znotraj Android Studia smo uporabili še paket Android SDK verzije 26.0.0.

4.3.2 Visual Studio

Microsoft Visual Studio (krajše Visual Studio) je integrirano razvojno okolje (IDE), ki ga je razvilo podjetje Microsoft [43]. Namenjen je razvijalcem programske opreme za razvoj namiznih aplikacij, spletnih aplikacij in spletnih storitev. Nudi dobro podporo različnim programskim jezikom, med drugim C++, C#, F#, R in JavaScript. Vsebuje zbirko številnih orodij za pisanje in razhroščevanje programske kode, s podporo številnim vtičnikom omogoča razširljivost. Pri implementaciji programskega vmesnika API in aplikacijskega strežnika smo uporabili Visual Studio 2017, ki podpira .NET ogrodje verzije 4.7. Znotraj okolja Visual Studio omogoča tudi neposredno postavitev (angl. *deployment*) spletnih strani in storitev v oblak. Posledično s tem odpade vnaprejšnja konfiguracija celotnega strežniškega okolja. To funkcionalnost smo uporabili za postavitev programskega vmesnika REST API na storitveno platformo Azure.

4.3.3 Microsoft Azure

Microsoft Azure je odprta in prilagodljiva storitvena (angl. *software as a service*, krajše *Saas*) in infrastrukturna platforma (angl. *infrastructure as*

a service, krajše *IaaS*) v oblaku [44], razvita s strani podjetja Microsoft. Uporablja se za razvoj, postavitev in upravljanje aplikacij v globalni mreži podatkovnih centrov podjetja Microsoft. Microsoft Azure ponuja celo zbirko oblačnih storitev, vse od navideznih strojev, podatkovnih baz in storitev za hrambo podatkov do storitev za obdelavo podatkov, gostovanje spletnih ter mobilnih aplikacij.

Pri postavitvi aplikacijskega strežnika in pripadajočega programskega vmesnika API smo uporabili storitev za aplikacije (angl. *app service*), ki spada pod storitve SaaS. Storitev za aplikacije omogoča postavitev spletnih in mobilnih aplikacij ter programskih vmesnikov API, napisanih v kateremkoli od programskih jezikov .NET, .NET Core, Java, Ruby, Node.js, PHP ali Python. Za trajno hrambo podatkov smo izbrali storitev podatkovne baze Azure SQL, ki prav tako spada pod storitve SaaS.

4.4 Mobilne in spletne tehnologije

Pri razvoju mobilne aplikacije smo za komunikacijo z značkami NFC uporabili razvojni paket TapLinx ter knjižnico Retrofit za komunikacijo z aplikacijskim strežnikom preko programskega vmesnika REST API. Za razvoj strežniškega dela smo uporabili ogrodji ASP.NET Web API in Entity Framework ter podatkovno bazo Azure SQL in ogrodje za avtorizacijo OAuth 2.0.

4.4.1 TapLinx

TapLinx [45] je razvojni paket za platformo Android, ki ga je razvilo podjetje NXP in omogoča lažji in hitrejši razvoj aplikacij, ki pri svojem delovanju uporabljajo funkcionalnost NFC in pametne kartice oziroma značke proizvajalca MIFARE. Njegova glavna prednost je odprt javanski programski vmesnik API, ki enkapsulira (ogradi) ukaze za komunikacijo z brezstičnimi karticami družin MIFARE, NTAG in iCode. TapLinx je na voljo brezplačno, pred uporabo je obvezna le registracija na njihovi spletni strani, s katero pridobimo ključ za uporabo razvojnega paketa v aplikaciji. Deluje na napra-

vah z operacijskim sistemom Android 5.0 ali novejšim. V razvojni paket je vključena tudi dokumentacija vseh metod programskega vmesnika in nekaj primerov uporabe. Trenutna verzija razvojnega paketa TapLinx je 1.4 (izšla 16. novembra 2017), pri razvoju smo uporabili takrat aktualno verzijo 1.3.

4.4.2 Retrofit

Retrofit je odprtokodna knjižnica, ki jo je razvilo podjetje Square [46]. Ponuja ogrodje za avtentikacijo in interakcijo z oddaljenimi programskimi vmesniki REST API preko protokola HTTP. Retrofit preslika metode oddaljenega programskega vmesnika API v vmesnik Java. Tako vsaka metoda na vmesniku Java predstavlja eno oddaljeno metodo na vmesniku API, njena povratna vrednost (angl. *return value*) pa definira javanski razred, v katerega se bo odgovor iz vmesnika API deserializiral. Z anotacijami metode definiramo njeno preslikovanje v zahtevo HTTP. Pri razvoju smo uporabili verzijo 2.3.0.

4.4.3 ASP.NET Web API

Za razvoj programskega vmesnika REST API na strežniški strani smo izbrali ogrodje ASP.NET Web API [47], ki omogoča razvoj storitev HTTP za številne odjemalce, kot so spletni brskalniki in mobilne aplikacije. Ogrodje Web API je zgrajeno na osnovi ogrodja .NET in je v prvi vrsti namenjeno izdelavi storitev RESTful, ki odjemalcem navzven ponujajo svoje storitve in podatke. Podpira zahteve v formatih JSON ali XML in v obliki standardnih metod HTTP (POST, GET, PUT, PATCH in DELETE). Programski vmesnik REST API smo napisali v okolju Visual Studio, pri tem smo uporabili ogrodje ASP.NET Web API verzije 5.2.3.

4.4.4 Entity Framework

Entity Framework (EF) je ogrodje za objektno-relacijsko preslikovanje (ORM) za ogrodje .NET [48]. V preteklosti je bilo ogrodje EF na voljo kot standar-

den del ogrodja .NET, od verzije 6 naprej je na voljo samostojno. Z vpeljavo nivoja abstrakcije premošča vrzel med objektno usmerjenim okoljem .NET in med okoljem z relacijskimi podatkovnimi bazami. Posledično se razvijalcem v okolju .NET pri svojem delu s podatki ni treba ukvarjati s poizvedbami, relacijami ali z arhitekturo na nivoju podatkovne baze. Obstajata dva načina uporabe ogrodja. Prvi način je izdelava entitentnih razredov, medsebojnih relacij in preslikovanj na podlagi že obstoječega modela podatkovne baze. Drugi je kodni model najprej (angl. *code first*), ki omogoča izdelavo podatkovne baze, njenih tabel in relacij na podlagi objektno usmerjenega konceptualnega modela, napisanega v okolju .NET. Zadnja verzija ogrodja EF 6.2 je izšla 26. oktobra 2017, mi smo pri našem razvoju uporabili verzijo 6.1.3.

4.4.5 Podatkovna baza Azure SQL

Za shranjevanje podatkov o značkah NFC in oskrbovalnih verigah smo uporabili relacijsko podatkovno bazo Azure SQL (angl. *Azure SQL database*) [49], ki je bila postavljena v oblak Azure. Njena posebnost je ta, da za njeno delovanje ni treba posebej postaviti namenskih strežnikov SQL, temveč za vse to poskrbi storitev Azure za oblačno podatkovno bazo. Po vseh ostalih lastnostih se bistveno ne razlikuje od klasične relacijske podatkovne baze Microsoft SQL in samostojnega strežnika SQL. Za izdelavo tabel in pripadajočih relacij smo uporabili za to namensko integrirano razvojno okolje SQL Server Management Studio [50]. Orodje nam poleg izdelave entitetno-relacijskega modela omogoča tudi izvajanje in razhroščevanje poizvedb SQL ter postavitev in upravljanje s strežniki SQL, strežniki Azure SQL in podatkovnimi skladišči SQL. Pri izdelavi podatkovnega modela smo uporabili verzijo orodja SQL Server Management Studio 2017, verzija podatkovne baze na oblačni storitvi Azure je bila enaka 12.0.2000.8.

4.4.6 OAuth

V tradicionalnem načrtovalskem vzorcu odjemalec-strežnik mora odjemalec za dostop do zaščitenih virov posredovati strežniku uporabnikove prijavne podatke [51]. Da lahko tretje osebe (aplikacije, odjemalci virov) pridobijo dostop do zaščitenih virov, mora odjemalec deliti uporabnikove prijavne podatke s strežnikom. To predstavlja več težav; odjemalec mora hraniti uporabnikove prijavne podatke (tipično celo v nešifrirani obliki) za nadaljnjo uporabo; strežnik mora omogočati avtentikacijo z geslom, kljub temu da je temu načinu dokazana varnostna ranljivost; tretje osebe lahko pridobijo preširok dostop do zaščitenih virov, pri tem pa uporabnik ne more spreminjati nivoja pravic za dostop; uporabnik ne more odvzeti dovoljenja za dostop posamezni tretji osebi, ne da bi pri tem utrpele posledice vse tretje osebe; če je varnost tretje osebe ogrožena (uporabnikovo geslo postane znano), posledično to pomeni, da so ogrožene vse preostale tretje osebe in vsi uporabniški podatki zaščiteni s tem geslom.

Protokol OAuth premosti omenjene težave z uvedbo plasti s pooblastili (angl. *authorization layer*) in z delitvijo vloge odjemalca virov od uporabniške vloge. OAuth je odprti standard za avtorizacijo in omogoča tretjim osebam (odjemalcem virov), da pridobijo omejen dostop do storitev HTTP (npr. dostop do programskega vmesnika API) v imenu lastnika sredstev (končni uporabnik ali drug odjemalec) ali v svojem imenu [35]. Pri protokolu OAuth odjemalec virov zaprosi za dostop do zaščitenih virov v imenu uporabnika, pri tem pridobi drugačno množico prijavnih podatkov – namesto uporabnikovega uporabniškega imena in gesla pridobi žeton za dostop (angl. *access token*). Žeton za dostop je običajno niz znakov, ki ima dodeljeno veljavnost in nivo pravic za dostop. Iz žetona za dostop ni mogoče pridobiti podatkov o uporabniku (le strežnik, ki je izdal žeton, ve, komu pripada žeton). Odjemalec virov za dostop do zaščitenih virov preko protokola OAuth vedno uporabi žeton za dostop, saj se tako izmenjujejo le informacije o avtorizaciji, identiteta odjemalca (uporabniško ime in geslo) ostane skrita in se ne izmenjuje.

Protokol OAuth 1.0 je bil uradno definiran aprila leta 2010 kot zahteva za mnenja (angl. *Request for Comments*, krajše RFC) RFC 5849 [52]. Čeprav ima protokol OAuth 1.0 dobro idejo, se ni prenesel v prakso zaradi kompleksnosti pri implementaciji. Oktobra leta 2012 je bil uradno definiran protokol OAuth 2.0 kot RFC 6749 [51]. Njegov glavni namen je avtorizacija odjemalcev virov v imenu uporabnika za dostop do zaščitenih virov, kot so to na primer programski vmesniki API. Čeprav ni obratno združljiv (angl. *backward compatible*) s protokolom OAuth 1.0, velja za njegovega naslednika in ponuja različne načine avtorizacije za spletne, namizne ter mobilne aplikacije in ostale pametne naprave, ki se povezujejo s spletom. Protokol priporoča uporabo žetonov za dostop z omejeno veljavnostjo. Dodatno lahko strežnik za avtorizacijo poleg žetona za dostop izda tudi žeton za osvežitev (angl. *refresh token*), katerega odjemalec virov varno shrani. Po preteku veljavnosti žetona za dostop odjemalec virov pošlje tak žeton za osvežitev kot zahtevo za nov žeton za dostop.

Pri razvoju smo za avtorizacijo odjemalcev virov (v našem primeru je to le mobilna aplikacija), ki odstopajo do programskega vmesnika API, uporabili protokol OAuth 2.0. Vključili smo ga v ogrodje ASP.NET Web API. Deluje na principu žetona JWT (angl. *JSON web token*), ki spada pod standard RFC 7519.

Poglavje 5

Rezultati testiranja in analiza

Osrednji del magistrskega dela predstavlja razvoj treh modelov, ki omogočajo označevanje, spremljanje in preverjanje avtentičnosti izdelkov, opremljenih z značko NFC. Podrobneje si bomo pogledali delovanje razvitih rešitev in jih analizirali. Poglavje bomo zaključili z opisom primerov uporabe in možnih napadov na sistem.

Testiranje in analiza delovanja sta bila izvedena v istem testnem okolju. Uporabili smo mobilni telefon Nexus 5x z operacijskim sistemom Android 8.1 in brezžično internetno povezavo. Aplikacijski strežnik in podatkovna baza sta bila postavljena v oblačno storitev Azure. Vsa komunikacija med značko NFC in mobilno aplikacijo poteka preko metod razvojnega paketa TapLinx, ki temelji na programskem vmesniku Android NFC.

Testni primer smo zasnovali kot oskrbovalno verigo s petimi deležniki (proizvajalec – logistika – skladišče – logistika – prodajno mesto, Slika 3.2), kamor smo vpisovali značke NFC in jo poimenovali slovenski vinograd. Za testni izdelek pa smo za vse tri modele izbrali steklenico belega vina.

Pri vseh aktivnostih smo v izsekih kode zapisali dejanske podatke, ki se prenašajo med značko NFC, mobilno aplikacijo in aplikacijskim strežnikom ter se shranjujejo v podatkovno bazo. Izseke kode smo oblikovali v notaciji, podobni formatu JSON, saj se ta tip formata uporablja pri izmenjavi podatkov preko programskega vmesnika REST API.

Na koncu smo za vsak model izvedli tudi meritve časovnih zahtevnosti vseh aktivnosti. Pri tem smo izvzeli tiste dele aktivnosti, ki so odvisne od stanja internetne povezave ter delovanja spletnega strežnika in podatkovne baze. Meritve so tako zajemale le tisti del aktivnosti, ki je neodvisen od zunanjega okolja in se razlikuje glede na model ter ga je smiselno medsebojno primerjati – komunikacija med mobilnim telefonom in različnimi tipi značk NFC.

5.1 Vpis nove značke v sistem oskrbovalne verige

Prva aktivnost, ki smo jo analizirali, je vpis nove značke v sistem oskrbovalne verige, kjer se v ozadju izvedeta dva procesa. Vsak izmed njih dostopa do značke NFC in izvede klic na aplikacijski strežnik preko programskega vmesnika API.

Prvi proces se sproži, ko uporabnik približa mobilni telefon znački NFC. Mobilna aplikacija prebere UID in tip značke (polji *uid, type*) in ju pošlje na strežnik kot zahtevo za poizvedbo po že obstoječi znački. V zahtevi pošlje še ID uporabnika (*supplyChainActor*). Izsek kode 5.1 prikazuje vsebino zahteve za poizvedbo po obstoju značke za Model 3. Oblika zahteve je enaka za Model 1 in Model 2, razlika je v vsebini polj *uid* in *type*.

Izsek kode 5.1: Zahteva za poizvedbo po obstoju značke

```
1 {
2     "tag": {
3         "uid": "0x04707B6AE12A80",
4         "type": "DESFire EV1"
5     },
6     "supplyChainActor": {
7         "id": 15
8     }
9 }
```

Sledi drugi proces, kjer mobilna aplikacija oblikuje zahtevo za vpis nove značke NFC v sistem (Izsek kode 5.2), pred tem pa izvede še naslednje korake:

- Za vse tri modele ustvari naključen niz. Pri Modelu 1 se ta prenese na strežnik v čistopisu, kar je označeno kot *contentPlain*.
- Pri Modelu 2 in Modelu 3 izračuna še zgoščeno vrednost niza po algoritmu SHA-256 (*contentHashed*) ter ustvari naključne vrednosti za ključa DES (*desKey1, desKey2*) oziroma ključa AES (*aesMasterKey, aesAppKey*).
- Pri Modelu 3 pred zapisom niza ustvari novo aplikacijo na znački (njen identifikator pošlje kasneje v zahtevi kot *appId*) in tekstovno datoteko, kamor lahko v naslednjem koraku zapiše niz.
- V tem koraku zapiše niz na značko NFC ter pri Modelu 2 in Modelu 3 nastavi še zaščito pomnilnika pred nedovoljenim branjem in pisanjem.
- Vse zbrane podatke na koncu pošlje na strežnik kot zahtevo za vpis nove značke NFC v sistem. Vsaka zahteva vsebuje tudi ID izbrane oskrbovalne verige (*supplyChainId*) ter ime, opis in črtno kodo izdelka (*productName, productDescription, barcode*).

Izsek kode 5.2: Zahteva za vpis nove značke v sistem

```
1  // Model 1
2  {
3      "tag": {
4          "uid": "0x049C7EEAE84C80",
5          "type": "NTAG213",
6          "supplyChainId": 3,
7          "itemData": {
8              "productName": "White Wine 0.75L",
9              "productDescription": "Quality white wine.",
10             "barcode": "2501101530031"
11         },
```

```
12          "contentPlain": "Pp7pTFEa#1b:FzeLGsO108ZeABku$O&z?t!YI$eRbJ&
                oefZ5d.8ryy8vHh!;h;xgK801O#XuTKcuT#QMx&ndrBXhYsl2zgLSu?
                ZA?ZDeHcw3I5Q$1jZiP:luw4bTM:8uXg"
13      },
14      "supplyChainActor": {
15          "id": 15
16      }
17  }
18
19  // Model 2
20  {
21      "tag": {
22          "uid": "0x04AD2D4A8C2A80",
23          "type": "Ultralight C",
24          "supplyChainId": 3,
25          "itemData": {
26              "productName": "White Wine 0.75L",
27              "productDescription": "Quality white wine.",
28              "barcode": "2501101530031"
29          },
30          "contentHashed": "6b18f3786402945999efe2750b9ffc35d957f68dad997394aa
                56f189bff17ee4",
31          "desKey1": "TmC:t#f6",
32          "desKey2": "g6I6&b;c"
33      },
34      "supplyChainActor": {
35          "id": 15
36      }
37  }
38
39  // Model 3
40  {
41      "tag": {
42          "uid": "0x04707B6AE12A80",
43          "type": "DESFire EV1",
44          "supplyChainId": 3,
45          "itemData": {
46              "productName": "White Wine 0.75L",
```

```
47                "productDescription": "Quality white wine.",
48                "barcode": "2501101530031"
49            },
50            "contentHashed": "445c4f74a997b2faf6b7277758aa2c63c914611c2b3439ef9
                  cfc2b1fe8e1cfc4",
51            "aesMasterKey": "HD1BEXhW4g;yYI9r",
52            "aesAppKey": "ptKF2f?mqfVKw;SI",
53            "appId": "01"
54        },
55        "supplyChainActor": {
56            "id": 15
57        }
58    }
```

V pomnilnik značk NFC je mobilna aplikacija ob vpisu nove značke zapisala nize, ki so navedeni v Izseku kode 5.3, ločeni po modelih. Pri Modelu 1 in Modelu 2 sta niza zapisana v obliki sporočila NDEF, pri Modelu 3 je niz zapisan v obliki standardne podatkovne datoteke DESFire (angl. *DESFire standard data file*).

Izsek kode 5.3: Naključni nizi, zapisani na značke NFC

```
1  // Model 1
2  Pp7pTFEa#1b:FzeLGsO108ZeABku$O&z?t!YI$eRbJ&oefZ5d.8ryy8vHh!;h;xgK
       801O#XuTKcuT#QMx&ndrBXhYsl2zgLSu?ZA?ZDeHcw3I5Q$1jZiP:luw4
       bTM:8uXg
3
4  // Model 2
5  L#Ctc3IqV6vtHx0eSn$5nYWcnSVlvcD:xP?D3?vldg.LA9bSPPFXNiHsgnTgV9bl
       5!9nbnypd0ki&uLqU#rh8x3835jHYnEs4&aH40Yj6BGj?WQ0GTFgs#$i8PIy7
       GNsmU
6
7  // Model 3
8  PXVr2YAzNd7yxm8qQm2ytWZY;W7w.hV0.f2;G:wIrUMucTbP;b4rjUZd#Ois&Cd
       ;J&&DhJdmygUZ9ec4T;l1Qa9inq&zQHxCVK0epgA3uj4d;TQU525G#yQ?A4
       XX.:FkEm
```

5.2 Dodajanje novega zapisa gibanja značke v oskrbovalno verigo

Dodajanje novega zapisa gibanja značke v oskrbovalno verigo za vse modele poteka na enak način in je razdeljeno na dva procesa.

Prvi proces se prične izvajati v trenutku, ko uporabnik približa svoj mobilni telefon znački NFC in mobilna aplikacija prebere UID značke. Aplikacija ga nato skupaj z ID uporabnika pošlje na aplikacijski strežnik kot poizvedbo po podatkih o izdelku in uporabniku (Izsek kode 5.4).

Izsek kode 5.4: Poizvedba po podatkih o izdelku in uporabniku

```
{
    "tag": {
        "uid": "0x049C7EEAE84C80"
    },
    "supplyChainActor": {
        "id": 15
    }
}
```

Strežnik ob prejemu zahteve izvede poizvedbo v podatkovni bazi in vrne mobilni aplikaciji vse zahtevane podatke, skupaj z ID in nazivom pripadajoče oskrbovalne verige (*supplyChainId, supplyChainTitle*) ter ID in nazivom vloge uporabnika v tej verigi (*supplyChainRoleId, supplyChainRoleTitle*) (Izsek kode 5.5).

Izsek kode 5.5: Odgovor strežnika na poizvedbo po podatkih o izdelku in uporabniku

```
{
    "tag": {
        "id": 25,
        "uid": "0x049C7EEAE84C80",
        "supplyChainId": 3,
        "supplyChainTitle": "Slovenian Vineyard",
        "itemData": {
```

```
8                "productName": "White Wine 0.75L",
9                "productDescription": "Quality white wine.",
10               "barcode": "2501101530031"
11           }
12       },
13       "supplyChainActor": {
14           "id": 15,
15           "supplyChainId": 3,
16           "supplyChainTitle": "Slovenian Vineyard",
17           "supplyChainRoleId": 13,
18           "supplyChainRoleTitle": "Logistics Company"
19       }
20   }
```

Drugi proces se izvede, ko uporabnik izbere ustrezen tip dogodka (dogodek odpreme oziroma prevzema) in pritisne na gumb za dodajanje zapisa. Mobilna aplikacija pošlje na strežnik UID značke, ID uporabnika ter izbran tip dogodka (*recordType*) (Izsek kode 5.6).

Izsek kode 5.6: Zahteva po vpisu novega zapisa gibanja značke

```
1   {
2       "tag": {
3           "uid": "0x049C7EEAE84C80"
4       },
5       "supplyChainActor": {
6           "id": 15
7       },
8       "supplyChainMovement": {
9           "recordType": "dispatch"
10      }
11  }
```

5.3 Preverjanje avtentičnosti značke

Pri preverjanju avtentičnosti se v ozadju izvedeta dva procesa, kjer v vsakem izmed njih dostopamo do značke in izvedemo klic na programski vmesnik

API. Pri Modelu 1 se izvede le drugi proces za branje vsebine pomnilnika značke NFC in pošiljanje zahteve za preverjanje avtentičnosti značke.

Prvi proces se prične, ko mobilna aplikacija prebere UID značke in ga pošlje na strežnik kot poizvedbo po ključih DES (Model 2) oziroma ključih AES (Model 3) za dostop do pomnilnika značke (Izsek kode 5.7).

Izsek kode 5.7: Poizvedba po ključih DES oz. AES – Model 2 in Model 3

```
1  // Model 2
2  {
3      "tag": {
4          "uid": "0x04AD2D4A8C2A80",
5          "type": "Ultralight C"
6      }
7  }
8
9  // Model 3
10 {
11     "tag": {
12         "uid": "0x04707B6AE12A80",
13         "type": "DESFire EV1"
14     }
15 }
```

Strežnik ob prejemu poizvedbe po ključih naredi naslednje:

- Mobilni aplikaciji vrne pripadajoče ključe – Model 2 (dva ključa DES), Model 3 (aplikacijski ključ AES in ID aplikacije).
- Pri Modelu 3 v odgovor doda še modul n in šifrirni eksponent e (kodirana po shemi Base64), oba predstavljata njegov javni ključ RSA. Razlog za pošiljanje modula in šifrirnega eksponenta je v povezljivosti (angl. *interoperability*) strežniške logike, napisane v programskem jeziku C#, in mobilne aplikacije, napisane v programskem jeziku Java. Na ta način pri obeh dosežemo povsem enako delovanje algoritma RSA. Modul in šifrirni eksponent sta v osnovi definirana kot polji bajtov (angl. *byte array*), vendar ju zaradi lažje predstavitve v obliki niza in

lažjega prenosa preko programskega vmesnika API kodiramo po shemi Base64.

Vsebina odgovora strežnika s ključi za dostop do pomnilnika značke in s strežnikovim javnim ključem je prikazana v Izseku kode 5.8.

Izsek kode 5.8: Odgovor strežnika s ključi DES oz. AES – Model 2 in Model 3

```
1  // Model 2
2  {
3      "tag": {
4          "uid": "0x04AD2D4A8C2A80",
5          "desKey1": "TmC:t#f6",
6          "desKey2": "g6I6&b;c"
7      }
8  }
9
10 // Model 3
11 {
12     "tag": {
13         "uid": "0x04707B6AE12A80",
14         "aesAppKey": "ptKF2f?mqfVKw;SI",
15         "appId": "01"
16     },
17     "rsaCryptoData": {
18         "n": "sCoXCBkQQoJvWKHTa2MJvQK83lGy4LkX0ljwjE4
               GFVgQyBQMMkAul7mvbB/8GKvjOvePNYth2+D2j3
               WTwkOEZaEjupsnz0CMB1a0G1io3MMJzy3Nxkbr4+BOPFW0
               uRYUUkYNPE3rQp6ylRas9N+opeAiEHiFRIeKE1/
               MLCtRQQBNpPo+acmAGNOf4ReBCGZk/Uk/lhKfN++lgb294OP2
               LBualXW/y62kMmDUvxsau+tyZoH6EUDagdptL6Z0YY55DJ5
               PGrKbDF/hSRe5SqvDNCUbu1K3gDTZl9TZrAIpc8V2b2oyqNvj1
               wMgTl6D/HW1J2ESGX5Gmm5ab/JDW8g4QQ==",
19         "e": "NjU1Mzc="
20     }
21 }
```

Sledi drug proces, ko mobilna aplikacija dostopa do pomnilnika značke NFC in prebere celotno vsebino, kar predstavlja naključen niz. Pri Mo-

delu 2 in Modelu 3 aplikacija pred branjem uporabi prejeta ključa za dostop iz prejšnjega procesa ter nato za prebran niz izračuna zgoščeno vrednost po algoritmu SHA-256. Pri Modelu 3 dobljeno zgoščeno vrednost šifrira s strežnikovim javnim ključem RSA in jo nato še kodira po shemi Base64. V zadnjem koraku mobilna aplikacija prebran niz iz pomnilnika značke oziroma njegovo zgoščeno vrednost pošlje na strežnik kot zahtevo po preverjanju avtentičnosti. Zadnji parameter – uporabnikov ID je opcijski. Mobilna aplikacija ga pošlje v zahtevi le, če je uporabnik prijavljen v aplikacijo (Izsek kode 5.9).

Izsek kode 5.9: Zahteva po preverjanju avtentičnosti

```
1  // Model 1
2  {
3      "tag": {
4          "uid": "0x049C7EEAE84C80",
5          "type": "NTAG213",
6          "contentPlain": "Pp7pTFEa#1b:FzeLGsO108ZeABku$O&z?t!YI$eRbJ&
               oefZ5d.8ryy8vHh!;h;xgK801O#XuTKcuT#QMx&ndrBXhYsl2zgLSu?
               ZA?ZDeHcw3I5Q$1jZiP:luw4bTM:8uXg"
7      },
8      "supplyChainActor": {
9          "id": 15
10     }
11 }
12
13 // Model 2
14 {
15     "tag": {
16         "uid": "0x04AD2D4A8C2A80",
17         "type": "Ultralight C",
18         "contentHashed": "6b18f3786402945999efe2750b9ffc35d957f68dad997394aa
               56f189bff17ee4",
19     }
20 },
21 "supplyChainActor": {
22     "id": 15
```

```
23 }
24
25 // Model 3
26 {
27     "tag": {
28         "uid": "0x04707B6AE12A80",
29         "type": "DESFire EV1",
30         "contentHashCiphered": "S4YKuSHdZUFSc6G4KmrRcxPuu56Y874U/
               lBoXKl/FQkdr1qD3BBNoWakTbV9Mks103
               DMWiWJYxRIIpmKfjGLnZ/nLkpHLwKfhAwGeCKaT0wzOkAkon/
               tyvTU6Kjeky0QfXjriJVru6U1VGVorZn5UKM1PdwbL5hqCcztlFtP1
               CI7Afk8sH17ftE8s7xa8x6BaAv0U6z1p9IyeK6s2UTH0lwpaDX53
               xkenCKhYWotV1Q7LNYpyhqoNoDw2HTQQhhCZ1OJ0VHXBr0
               VFZfOGRdZzkswNbPOBugVU1PTz8rltbqiWRBFeqYQJE7GgwtK29
               QUBbTIgMSw+1D1q3Sfm+bMsw==",
31     },
32     "supplyChainActor": {
33         "id": 15
34     }
35 }
```

Strežnik na podlagi prejetih podatkov in pridobljenih zapisov iz podatkovne baze izvede preverjanje avtentičnosti. Mobilni aplikaciji nato vrne ustrezno potrdilo oziroma zavrnitev avtentičnosti (*isAuthentic*). Če je zahtevo za preverjanje poslal prijavljen uporabnik, mu strežnik v odgovoru vrne tudi seznam vseh zapisov gibanja značke v oskrbovalni verigi (*scmRecordList*). Odgovor je pri vseh modelih enake oblike (Izsek kode 5.10).

Izsek kode 5.10: Potrdilo avtentičnosti in opcijski zapisi gibanja značke

```
1 {
2     "tag": {
3         "uid": "0x04AD2D4A8C2A80",
4         "supplyChainId": 3,
5         "itemData": {
6             "productName": "White Wine 0.75L",
7             "productDescription": "Quality white wine.",
8             "barcode": "2501101530031"
```

```
        },
        "isAuthentic": "true"
    },
    "scmRecordList": [{
            "id": 30,
            "recordAddedOn": "2017-09-10T20:50:39.443",
            "type": "enrollment",
            "userName": "admin@authentitap.com",
            "supplyChainActorId": 1,
            "supplyChainRoleTitle": "Wine Manufacturer"
        }, {
            "id": 31,
            "recordAddedOn": "2017-09-10T20:50:46.717",
            "type": "dispatch",
            "userName": "admin@authentitap.com",
            "supplyChainActorId": 1,
            "supplyChainRoleTitle": "Wine Manufacturer"
        }, {
            "id": 32,
            "recordAddedOn": "2017-09-10T20:51:13.743",
            "type": "receive",
            "userName": "user1@logistics.com",
            "supplyChainActorId": 3,
            "supplyChainRoleTitle": "Distributor Logistics"
        }, {
            "id": 33,
            "recordAddedOn": "2017-09-10T20:51:42.423",
            "type": "dispatch",
            "userName": "user1@logistics.com",
            "supplyChainActorId": 3,
            "supplyChainRoleTitle": "Distributor Logistics"
        },
        ...
    ]
}
```

5.4 Delovanje aplikacije – časovne zahteve

Za vsako aktivnost sistema smo izvedli tudi meritve časovnih zahtevnosti tistih delov procesa, ki so neodvisni od zunanjega okolja in se razlikujejo glede na model. Razlike med modeli se pojavijo zaradi drugačnega načina dostopa do značk NFC ter branja in pisanja vsebine v pomnilnike značk. Pri vsakem modelu smo meritve izvedli 50-krat, jih programsko izmerili znotraj izvajanja kode (Izsek kode 5.11) in na koncu izračunali njihovo povprečno trajanje.

Izsek kode 5.11: Primer programskega časovnega merjenja – branje vsebine pomnilnika značke NFC

```
1  final long startTime = System.nanoTime();
2  String tagContent = tag.ReadContent();
3  long endTime = System.nanoTime();
4
5  // Branje pomnilnika je v milisekundah trajalo:
6  double deltaMs = (endTime − startTime) / 1e6;
```

Vpis nove značke v sistem oskrbovalne verige

Pri vpisu nove značke v sistem nas je zanimal tisti del procesa, kjer upravljamo s pomnilnikom značke NFC in vanj zapisujemo naključno ustvarjen niz. V Tabeli 5.1 smo zapisali povprečne vrednosti izmerjenih meritev.

Pri izračunu povprečnih vrednosti meritev smo ugotovili, da ni bistvenih razlik za Model 1 in Model 2. Dostop do pomnilnika značke NFC je pri obeh enak, enako morata oba pred zapisom niza v pomnilnik značke tega oblikovati v obliki sporočila NDEF. Zanimivo je, da pri Modelu 2 nastavljanje ključev znački NFC ne doprinese veliko k povprečnemu času. Večja razlika se pojavi pri Modelu 3, kjer upravljanje s pomnilnikom značke in zapisovanje niza traja do 54 % dlje kot pri Modelu 1 oziroma do 43 % dlje kot pri Modelu 2. Razlog za daljši povprečni čas pri Modelu 3 se skriva v dodatnem formatiranju pomnilnika značke NFC in oblikovanju datotečnega sistema.

Zapis vsebine na značko NFC	Model 1	Model 2	Model 3
Nastavitev ključev		x	x
Formatiranje pomnilnika			x
Zapis niza v pomnilnik	x	x	x
Povprečen čas	700 ms	754 ms	1078 ms
Standardni odklon	65 ms	76 ms	43 ms
min	619 ms	663 ms	1025 ms
max	827 ms	850 ms	1155 ms

Tabela 5.1: Povprečno trajanje upravljanja pomnilnika značke in zapisovanja niza v pomnilnik značke pri vpisu nove značke v sistem

Izmed 150 izvedenih aktivnosti vpisa nove značke v sistem se je pet izmed njih končalo z napako, kar predstavlja 97-% uspešnost. Razlog za napako se najverjetneje skriva v prezgodnjem odmiku mobilnega telefona od značke NFC. Značka in mobilni telefon namreč morata ostati v obsegu delovanja, vse dokler aktivnost vpisa ni v celoti zaključena, drugače vpis ne bo uspešen.

Trajanje celotne aktivnosti vpisa značke v sistem, ki zajema oba procesa, smo na enak način programsko izmerili in izračunali njeno povprečje po izvedenih 50 meritvah za vsak model. Tu naj opomnimo, da so v meritvah zajeti tudi vsi zunanji dejavniki (stanje internetne povezave, delovanje aplikacijskega strežnika in podatkovne baze), zato so povprečne vrednosti zgolj informativne (Tabela 5.2).

Vpis značke NFC v sistem	Model 1	Model 2	Model 3
Povprečen čas	1100 ms	1230 ms	2320 ms

Tabela 5.2: Povprečno trajanje celotnega vpisa nove značke v sistem

Dodajanje novega zapisa gibanja značke v oskrbovalno verigo

Aktivnost dodajanja novega zapisa gibanja značke se glede na model razlikuje le v prvem procesu, ko mobilna aplikacija bere tip in UID značke. Celotno aktivnost smo za vsak model izvedli 50-krat, pri tem pa programsko izmerili trajanje prvega procesa in nato izračunali povprečne vrednosti (Tabela 5.3).

Branje tipa in UID značke	Model 1	Model 2	Model 3
Dostop do značke	x	x	x
Povprečen čas	3,72 ms	4,61 ms	4,65 ms
Standardni odklon	1,37 ms	0,90 ms	1,39 ms
min	1,71 ms	3,16 ms	2,70 ms
max	6,69 ms	6,44 ms	7,27 ms

Tabela 5.3: Povprečno trajanje branja tipa in UID značke pri dodajanju novega zapisa gibanja značke

Razlike v povprečnih vrednostih med modeli so zanemarljive. Pri vseh izvedenih 150 poskusih dodajanja novega zapisa gibanja značke se je aktivnost uspešno zaključila, kar predstavlja 100-% uspešnost. Razlog za popolno uspešnost pripisujemo naravi samega procesa, saj branje tipa in UID značke poteka hitro. Posledično se zmanjša prostor za uporabniške napake, kjer bi uporabniku v tem kratkem času uspelo umakniti mobilni telefon od obsega delovanja značke.

Pri vseh modelih se je celotna aktivnost v poprečju izvedla v 530 ms. V tem času ni zajeta uporabniška interakcija v drugem procesu, ko uporabnik izbere tip dogodka in pritisne gumb za potrditev.

Preverjanje avtentičnosti značke

Za vsak model smo izvedli tudi 50 poskusov aktivnosti preverjanja avtentičnosti značke in izmerili ter izračunali povprečno trajanje tistega dela, ki se razli-

kuje glede na model. Upoštevan je del drugega procesa, ko mobilna aplikacija dostopa do značke (vključno s protokolom za avtentikacijo pri Modelu 2 in Modelu 3), bere vsebino pomnilnika značke ter nato izračuna zgoščeno vrednost prebranega niza in jo šifrira. V Tabeli 5.4 so predstavljene izračunane povprečne vrednosti trajanja.

Branje vsebine značke, dodatne operacije	Model 1	Model 2	Model 3
Dostop do značke	x	x	x
Branje vsebine pomnilnika značke	x	x	x
Izračun zgoščene vrednosti niza		x	x
Šifriranje zgoščene vrednosti niza			x
Povprečen čas	172 ms	191 ms	252 ms
Standardni odklon	14 ms	15 ms	22 ms
min	153 ms	174 ms	215 ms
max	203 ms	228 ms	294 ms

Tabela 5.4: Povprečno trajanje branja vsebine pomnilnika ter izvedbe dodanih operacij nad prebranimi podatki pri preverjanju avtentičnosti značke

Glede na povprečne vrednosti je Model 3 v primerjavi z Modelom 1 do 47 % počasnejši pri dostopu in branju vsebine pomnilnika značke. Razlika se pojavi zaradi dodatnega računanja zgoščene vrednosti prebranega niza in njegovega šifriranja. V primerjavi z Modelom 2 pa ta razlika za Model 3 znaša 32 %.

Med 150 izvedenimi poskusi preverjanj avtentičnosti značke se 17 izmed njih ni končalo uspešno, kar predstavlja 89-% uspešnost. Razlog za slabšo uspešnost pripisujemo naravi procesa, saj mora uporabnik po približanju mobilnega telefona znački oba držati skupaj pri miru, vse dokler aktivnost ni v celotni zaključena. Obenem pa ne izključujemo možnosti, da je do napake prišlo pri prenosu vsebine iz mobilnega telefona na aplikacijski strežnik.

Za informativne namene smo izmerili tudi čas celotne aktivnosti preverjanja avtentičnosti za vsak model in izračunali povprečne vrednosti (Ta-

bela 5.5).

Preverjanje avtentičnosti	Model 1	Model 2	Model 3
Povprečen čas	455 ms	731 ms	2395 ms

Tabela 5.5: Povprečno trajanje celotnega preverjanja avtentičnosti značke

Glavni razlog za bistveno daljši povprečni čas pri Modelu 3 se skriva v generiranju strežnikovega javnega in zasebnega para ključa RSA, ki se izvede na strežniku v prvem procesu. Model 1 dosega v primerjavi z Modelom 2 in Modelom 3 boljše povprečne čase zaradi enega klica na aplikacijski strežnik. Pri osnovnem modelu namreč odpade prvi klic na strežnik z zahtevo po ključih značke.

5.5 Primeri uporabe

Vse obravnavane značke NFC so dobavljive v obliki obeskov, nalepk ali drugih načinih izvedbe, s katerimi je mogoče opremiti izdelke ali njihove embalaže. Na izbiro modela za zaščito izdelka, ki ga želimo opremiti z značko NFC, vplivajo razlike v cenah značk in v različnih stopnjah varnosti modelov. V Tabeli 5.6 so zapisane cene, ki veljajo ob nakupu posamezne oziroma večje količine značk NFC.[1]

Za izdelke nižjega cenovnega razreda in izdelke široke potrošnje je smiselna uporaba značk NTAG213 (Model 1). Z okvirno ceno deset centov to predstavlja upravičen strošek za proizvajalca glede na ceno izdelka, ki ga ščiti Model 1. Njegova uporaba je smiselna v takih oskrbovalnih verigah,

[1]Cene značk NFC so bile pridobljene 20. 1. 2018 iz spletne strani podjetja Avnet, ki ga NXP Semiconductors navaja kot uradnega proizvajalca njihovih proizvodov. Navedene cene veljajo pri nakupu ene ali večje količine značk in ne vsebujejo davka na dodano vrednost. Stran je dostopna na naslovu `https://www.avnet.com/wps/portal/emea`. Direktna povezava do značk: NTAG213, MIFARE Ultralight C, MIFARE DESFire EV1 2K.

Cena značke ob nakupu	NTAG213	MIFARE Ultralight C	MIFARE DESFire EV1 2K
1 kos	0,28 EUR	0,36 EUR	1,32 EUR
100 kosov	0,11 EUR	0,28 EUR	1,12 EUR
1000 kosov	0,10 EUR	0,27 EUR	1,07 EUR

Tabela 5.6: Cene značk

kjer sta sledljivost izdelkov in hitrost izvajanja aktivnosti pomembnejša od visoke stopnje zaščite izdelkov. Z nalepkami NTAG213 bi opremili:

- posamezen kos sadja ali zelenjave,
- plastenke pijač,
- droben pisarniški material.

V tem primeru bi naštetim izdelkom zagotovili popolno sledljivost ter osnovno zaščito avtentičnosti.

Za izdelke srednjega cenovnega razreda in za izdelke, ki so potencialna tarča ponarejevalcev, je primerna značka MIFARE Ultralight C (Model 2). Cena značke je malenkost višja kot pri znački NTAG213, vendar to doprinese k varnejšemu sistemu, ki ga nudijo značke MIFARE Ultralight C v povezavi z Modelom 2. Hitrost izvajanja se pri vpisu novih značk in preverjanju avtentičnosti bistveno ne podaljša v primerjavi z Modelom 1. Uporaba te rešitve se nam zdi primerna za naslednje izdelke:

- Zgoščenke z glasbeno ali video vsebino ter programsko opremo. Na njihove ovitke bi enostavno prilepili značke NFC v obliki nalepk.
- Prehranski izdelki in pijače pri katerih je pomembna zaščita avtentičnosti in porekla – mesni izdelki in ribe ter oljčno olje, vina in žgane pijače. Na embalažo oziroma steklenice bi enostavno prilepili značke NFC v obliki nalepk.

Model 3 je primeren za izdelke višjega cenovnega razreda, kjer se pogosto pojavljajo ponarejeni izdelki na tržišču. Značke MIFARE DESFire EV1,

ki jih model uporablja, stanejo okoli enega evra, kar predstavlja upravičen strošek za proizvajalca v zameno za visoko stopnjo zaščite njegovih izdelkov. Preverjanje avtentičnosti značke je pri tem modelu počasnejše kot pri Modelu 1 in Modelu 2. Glavni razlog za to je v dodatnem šifriranju RSA, ki zagotavlja integriteto in zaupnost podatkov, ki se prenašajo med mobilno aplikacijo in aplikacijskim strežnikom. Uporabo Modela 3 predlagamo za naslednji kategoriji izdelkov:

- Farmacevtski, kozmetični in prehrambeni izdelki, za katere je pomembna popolna sledljivost in preverljiva avtentičnost.
- Izdelki, kjer je zaščita intelektualne lastnine poglavitna – sestavni deli za avtomobilsko in letalsko industrijo, izdelki zabavne in računalniške elektronske opreme ter oblačila in obutev priznanih blagovnih znamk. Vsem naštetim izdelkom je skupna višja cena in visoko povpraševanje kupcev na trgu.

5.6 Možni napadi

Prototip sistema, ki smo ga predstavili, ni popolnoma imun na zunanje napade. Vseh splošnih in usmerjenih napadov je veliko, zato bomo skušali opisati in analizirati le tiste, ki se pojavljajo najpogosteje in predstavljajo resno grožnjo pri uporabi tehnologije NFC in delovanju strežnika [5, 53, 54].

Prvi možni napad, ki se lahko zgodi, je ohromitev storitve (angl. *denial of service*). Napadalec s pošiljanjem velikega števila zahtev iz mobilne aplikacije na programski vmesnik API ohromi delovanje aplikacijskega strežnika, tako da za druge uporabnike postanejo storitve strežnika nedostopne. Napad je možen pri vseh treh modelih. Preprečili oziroma omejili bi ga z implementacijo blokade oziroma zaklepa na aplikacijskem strežniku (angl. *lockout*). Ta uporabniku, ki je večkrat napačno vnesel prijavne podatke, onemogoči prijavo in izvajanje storitev za vnaprej določeno časovno obdobje. Prav tako bi bilo smiselno omejiti število klicev programskega vmesnika, ki jih nek uporabnik lahko izvede v določenem časovnem obdobju. Omenjeni izboljšavi bi

verjetnost za uspešno izveden napad z ohromitvijo storitve zmanjšali.

Napad z mavričnimi tabelami (angl. *rainbow table attack*) napadalec izvede, kadar dobi dostop do zgoščenih vrednosti gesel. V našem primeru so predvsem na udaru shranjene zgoščene vrednosti vsebine značk v podatkovni bazi. Napadalec lahko z vnaprej izračunanimi zgostitvami za najbolj pogoste nize poskuša najti ujemanje z zgostitvijo v podatkovni bazi in s tem pridobiti izvirno vrednost zgoščene vsebine. Napad je možen le pri Modelu 2, kjer shranjujemo zgoščene vrednosti brez uporabe soli. Pri Modelu 3 pa smo z uporabo soli v zgoščevalni funkciji tak napad omejili.

Ker se podatki med značko NFC in bralnikom NFC prenašajo brezžično na daljavo, napadalec lahko izvede napad s prisluškovanjem (angl. *eavesdropping*) in na ta način poskuša zajeti njuno komunikacijo. Z analizo zajetih podatkov potem skuša ugotoviti, kakšne podatke mora zapisati v pomnilnik značke duplikata, da bralnik ne bo znal razpoznati med izvirno in ponarejeno značko. Omenjen napad je možen pri Modelu 1 in Modelu 2, saj se podatki med značko NFC in mobilnim telefonom prenašajo v nezaščiteni obliki. Značka DESFire EV1 pri Modelu 3 pa omogoča, da se vsi podatki med njima prenašajo v šifrirani obliki.

Pri uporabi značk NFC je vedno možen napad kloniranja UID značk in posledično lažno predstavljanje. Da bi se temu napadu izognili, je poleg identifikacije značke NFC zgolj na podlagi njenega UID treba implementirati še dodatno preverjanje glede na druge lastnosti oziroma atribute značke. Mi smo v naših modelih zapisali naključen niz v pomnilnik značke, ki samo skupaj z UID enolično določa značko. Proizvajalec NXP Semiconductors za svoje značke trdi, da njihove UID ni mogoče spreminjati ali kakorkoli potvarjati. Napad kloniranja značk NFC je najbolj verjeten pri Modelu 1, saj sta UID in vsebina pomnilnika značke dostopna in berljiva vsakemu bralniku NFC, ki se nahaja v območju delovanja značke. Pri Modelu 2 in Modelu 3 pa bi napadalec moral pred branjem vsebine pomnilnika značke najprej razbiti protokol 3DES oziroma AES za avtentikacijo.

Zadnji napad, ki predstavlja potencialno grožnjo sistemu, je napad s po-

srednikom (angl. *man in the middle*). Napadalec se pri tem napadu postavi med značko NFC in bralnikom NFC in se obema lažno predstavi. Znački se predstavi kot legitimen bralnik NFC, bralniku pa kot legitimna značka. Vso komunikacijo med njima prestreže in s tem dobi možnost aktivnega spreminjanja podatkov. Podobno kot pri napadu s prisluškovanjem je verjetnost za uspešno izveden napad s posrednikom visoka za Model 1 in Model 2, pri Modelu 3 pa je zaradi uporabe šifrirane komunikacije tak napad težje izvedljiv. Napadalec bi moral namreč pred analizo zajetih podatkov ugotoviti šifrirni ključ AES, šele nato bi jih lahko spremenil ter ponovno šifriral in poslal znački oziroma bralniku NFC.

5.7 Analiza SWOT

Analiza SWOT (angl. *SWOT analysis*) je strukturirana metoda za analizo in ovrednotenje prednosti, slabosti, priložnosti in nevarnosti nekega projekta, poslovnega načrta ali sistema [55]. Z njo se prepozna in opredeli notranje in zunanje dejavnike, ki pozitivno ali negativno vplivajo na doseganje zastavljenih ciljev. Pod notranje dejavnike spadata kategoriji prednosti in slabosti, na katere imamo neposreden vpliv in izhajata neposredno iz predmeta vrednotenja. Priložnosti in nevarnosti pa so povezane z zunanjimi dejavniki in okoliščinami, na katere nimamo vpliva oziroma je ta omejen. V analizi se upošteva tudi časovni vidik – prednosti in slabosti se nanašajo na pretekli in sedanji čas, priložnosti in nevarnosti pa na prihodnost. Bistvo analize je prepoznava potencialov in nevarnosti, na podlagi katerih lahko lažje oblikujemo načrt ter izvedemo dodatne ukrepe. Za prototip sistema, ki vključuje vse tri modele, smo izdelali analizo SWOT in jo predstavili v obliki matrike v Tabeli 5.7.

Prednosti

Vsi trije modeli z uporabo mobilne aplikacije omogočajo enostaven način preverjanja avtentičnosti izdelka, ki je opremljen z značko NFC in je vključen

Prednosti	**Slabosti**
- preprosta uporaba in preverjanje avtentičnosti izdelka - zanesljivo delovanje - možna vključitev enega izmed modelov v obstoječ sistem	- stalna internetna povezava - na voljo za mobilne telefone s sistemom Android 5.0 ali novejšim - prosto dostopni podatki, zapisani na znački (Model 1)
Priložnosti	**Nevarnosti**
- preprosta zaščita izdelkov - enostaven sistem sledljivosti - velik trg izdelkov višjega cenovnega razreda - možnosti za izboljšave (Model 2 in Model 3)	- vdori in napadi - varnost sistema temelji na varnosti protokolov za avtentikacijo - majhna konkurenčna prednost

Tabela 5.7: Analiza SWOT za prototip sistema

v sistem oskrbovalne verige. Upravljanje z mobilno aplikacijo je preprosto, ugotovili pa smo tudi, da je njeno delovanje zanesljivo na manjšem številu izvedenih testov. Prototip sistema je zasnovan tako, da ga je možno z minimalni spremembami vključiti v obstoječe proizvodne in distributerske procese. Cena značk NFC predstavlja glavni strošek pri uporabi enega izmed modelov in tako ekonomsko upraviči vpeljavo takega sistema za večino izdelkov, pri katerih se pogosto pojavljajo ponaredki na trgu.

Slabosti

Mobilna aplikacija deluje le, če ima stalno internetno povezavo – gre torej za povezan (angl. *online*) sistem in predstavlja nasprotje nepovezanemu (angl. *offline*) sistemu, kjer stalna internetna povezava ni nujna za samo delovanje. Pri razvoju mobilne aplikacije smo uporabili razvojni paket TapLinx in njegove programske vmesnike. Posledično je možno našo aplikacijo namesti le na naprave, ki imajo operacijski sistem Android 5.0 ali novejši. Slabost Modela 1 in značke NFC izvedbe NTAG213 je tudi v prosto dostopnem po-

mnilniku značke, saj lahko vsak bralnik NFC prebere njeno vsebino.

Priložnosti

Iz prototipa sistema smo razvili tri modele, ki iz uporabniške perspektive omogočajo preprosto zaščito izdelkov in njihovo sledljivost. Na trgu sicer že obstajajo podobne rešitve, vendar bi lahko z Modelom 2 in Modelom 3 vstopili na trg izdelkov višjega cenovnega razreda. Kot priložnost, ki jo vidimo za naš sistem, so izboljšave za Model 2 in Model 3. Pri slednjem bi lahko namesto šifriranja RSA uporabili šifriranje na osnovi eliptičnih krivulj (ECC). Za doseganje primerljive varnosti bi se tako dolžina ključa skrajšala. Pri obeh modelih pa bi lahko dodatno šifrirali še ključe DES oziroma AES, ki se pošiljajo iz mobilne aplikacije na aplikacijski strežnik in obratno.

Nevarnosti

Kot največje zunanje grožnje, ki smo jih prepoznali za naš sistem, so napadi in vdori. Najpogostejše izmed njih smo že opisali in analizirali v prejšnjem poglavju. Najbolj varen izmed modelov je Model 3, kjer je vsa komunikacija med značko NFC, mobilno aplikacijo in aplikacijskim strežnikom šifrirana, zato se verjetnost za napade zmanjša. Ker pri zaščiti vsebine na znački za izhodišče uporabljamo proizvajalčev protokol za avtentikacijo, to posledično predstavlja najšibkejši člen v zaščiti avtentičnosti izdelkov. Proizvajalec NXP Semiconductors za svoje značke družine MIFARE in NTAG v preteklosti ni imel hujših varnostnih lukenj in dokazanih šibkosti, zato njegove značke danes veljajo za varne. S ceno dobrega evra na kos (Model 3) pa naš najmočnejši sistem ne more konkurirati podobnim rešitvam, kjer je ta cena nižja.

Poglavje 6

Sklepne ugotovitve

V magistrskem delu smo raziskali in preučili različne načine zagotavljanja avtentičnosti izdelkov in ugotavljanja ponaredkov. Iz danih izsledkov smo nato predstavili prototip sistema oskrbovalne verige in značk NFC. Izbrali smo take značke NFC, katere je možno enostavno dodati na izdelke ali vstaviti v njihove embalaže. Iz prototipa smo izpeljali tri konkretne modele, ki z uporabo različnih tipov značk NFC omogočajo različne stopnje varnosti in zaščite izdelkov. Izkazalo se je, da že z osnovnim modelom lahko zagotovimo primerno zaščito pred ponarejanjem, z najvarnejšim modelom pa dovolj trdno zaščito, ki je priporočljiva tudi za izdelke višjega cenovnega razreda. Potrošniku kot zadnjemu deležniku oskrbovalne verige smo omogočili, da pred nakupom izdelka preveri njegovo avtentičnost.

Čeprav smo poskušali zasnovati sistem čim bolj varno in uporabniku prijazno, še vidimo odprte možnosti za nadaljnje raziskave in izboljšave. Prva izboljšava sistema oskrbovalne verige, ki jo predlagamo, je integracija podatkovnega modela s standardiziranim katalogom izdelkov EPC ter z globalnim omrežjem za sinhronizacijo podatkov GS1 GDSN. V tem primeru bi imel administrator pri vpisu novih značk v sistem manj dela, saj bi že sama mobilna aplikacija pridobila ustrezne podatke in mu ponudila seznam izdelkov, ki so na voljo. Prav tako bi sledljivost izdelkov lahko preverjali tudi ostali deležniki, ki niso direktno vključeni v njihove oskrbovalne verige.

Druga možna izboljšava, ki jo vidimo, je sprotno preverjanje avtentičnosti izdelkov v oskrbovalnih verigah. Vsak deležnik bi tako imel možnost preveriti izvor in zgodovino izdelka ne glede na njegovo pozicijo v verigi. Na podlagi teh podatkov bi se potem odločil, ali naj prevzame oziroma zavrne izdelek, ki ga bo prejel od svojega predhodnika. Na ta način bi se verjetnost, da ponaredki preidejo v oskrbovalno verigo, precej zmanjšala.

Kot zadnjo izboljšavo predlagamo upoštevanje vrstnega reda deležnikov v oskrbovalni verigi pri dodajanju zapisov gibanja značke. V trenutni verziji mobilne aplikacije morajo vsi deležniki verige poznati in slediti vnaprej predpisanemu vrstnemu redu. Lahko pa se zgodi, da nek deležnik doda svoja dva zapisa (prevzem in odprema) prezgodaj ali prepozno glede na vrstni red. Posledično značka ne bo več avtentična. Aplikacija bi ob tej izboljšavi natančno upoštevala vrstni red deležnikov pri dodajanju zapisov in s tem preprečila morebitne uporabniške napake.

Predlagano rešitev zaščite izdelkov z uporabo tehnologije NFC in pametnih telefonov vidimo v oskrbovalnih verigah pri izdelkih višjega cenovnega razreda. Implementacija predstavlja nek dodaten strošek tako za razvoj aplikacije in nakup značk NFC, je pa njihova uporaba relativno enostavna in omogočena širokemu krogu uporabnikov.

Literatura

[1] OECD/EUIPO, Trade in Counterfeit and Pirated Goods: Mapping the Economic Impact, Dostopno na: http://dx.doi.org/10.1787/9789264252653-en (pridobljeno 17. 1. 2018).

[2] S. Charlebois, A. Schwab, R. Henn, C. W. Huck, Food fraud: An exploratory study for measuring consumer perception towards mislabeled food products and influence on self-authentication intentions, zv. 50, 2016, str. 211–218.

[3] M. Bolić, D. Simplot-Ryl, I. Stojmenović, RFID systems: Research Trends and Challenges, John Wiley & Sons Ltd., 2010.

[4] B. Fennani, H. Hamam, A. O. Dahmane, RFID overview, v zborniku: ICM 2011 Proceeding, 2011, str. 1–5.

[5] V. Coskun, K. Ok, B. Ozdenizci, Near Field Communication: From Theory to Practice, John Wiley & Sons Ltd., 2012.

[6] NFC.org, About Near Field Communication, Dostopno na: http://nearfieldcommunication.org/about-nfc.html (pridobljeno 8. 9. 2017).

[7] NFC Forum, Tag Type Technical Specifications, Dostopno na: http://nfc-forum.org/our-work/specifications-and-application-documents/specifications/tag-type-technical-specifications (pridobljeno 10. 9. 2017).

[8] NFC Forum Technical Specification, NFC Data Exchange Format (NDEF), Rev 1.0 Jul. 2006.

[9] NXP Semiconductors, About MIFARE, Dostopno na: https://www.mifare.net/en/about-mifare/ (pridobljeno 22. 9. 2017).

[10] NXP Semiconductors, Chip Card ICs, Dostopno na: https://www.mifare.net/en/products/chip-card-ics/ (pridobljeno 23. 9. 2017).

[11] MIFARE, MIFARE Classic Family, Dostopno na: https://www.mifare.net/en/products/chip-card-ics/mifare-classic/ (pridobljeno 22. 9. 2017).

[12] K. Nohl, D. Evans, S. Starbug, H. Plötz, Reverse-engineering a cryptographic rfid tag, v zborniku: Proceedings of the 17th Conference on Security Symposium, SS'08, USENIX Association, Berkeley, CA, USA, 2008, str. 185–193.

[13] NXP Semiconductors, MIFARE Plus EV1 - Product Short Data Sheet, Rev. 2.0, Dostopno na: https://www.nxp.com/docs/en/data-sheet/MF1P_H_X1Y1_SDS.pdf (pridobljeno 22. 9. 2017).

[14] NXP Semiconductors, MIFARE Plus SE - Product Short Data Sheet, Rev. 3.1, Dostopno na: https://www.nxp.com/docs/en/data-sheet/MF1SEP_H_10X1_SDS.pdf (pridobljeno 22. 9. 2017).

[15] NXP Semiconductors, MIFARE Ultralight Nano - Product Data Sheet, Rev. 3.1, Dostopno na: https://www.nxp.com/docs/en/data-sheet/MF0UN_H_00.pdf (pridobljeno 22. 9. 2017).

[16] NXP Semiconductors, MIFARE Ultralight EV1 - Product Data Sheet, Rev. 3.1, Dostopno na: https://www.nxp.com/docs/en/data-sheet/MF0ULX1.pdf (pridobljeno 22. 9. 2017).

[17] NXP Semiconductors, MIFARE Ultralight C - Product Data Sheet, Rev. 3.1, Dostopno na: `https://www.nxp.com/docs/en/data-sheet/MF0ICU2.pdf` (pridobljeno 22. 9. 2017).

[18] NXP Semiconductors, MIFARE DESFire EV1 - Product Short Data Sheet, Rev. 3.2, Dostopno na: `https://www.nxp.com/docs/en/data-sheet/MF3ICDX21_41_81_SDS.pdf` (pridobljeno 22. 9. 2017).

[19] NXP Semiconductors, MIFARE DESFire EV2 - Product Short Data Sheet, Rev. 2.0, Dostopno na: `https://www.nxp.com/docs/en/data-sheet/MF3DX2_MF3DHX2_SDS.pdf` (pridobljeno 22. 9. 2017).

[20] NXP Semiconductors, NTAG210/212 - Product Data Sheet, Rev. 3.0, Dostopno na: `https://www.nxp.com/docs/en/data-sheet/NTAG210_212.pdf` (pridobljeno 22. 9. 2017).

[21] NXP Semiconductors, NTAG213/215/216 - Product Data Sheet, Rev. 3.2, Dostopno na: `https://www.nxp.com/docs/en/data-sheet/NTAG213_215_216.pdf` (pridobljeno 22. 9. 2017).

[22] L. Li, Technology designed to combat fakes in the global supply chain, zv. 56, 2013, str. 167–177.

[23] M. Ghaani, C. A. Cozzolino, G. Castelli, S. Farris, An overview of the intelligent packaging technologies in the food sector, zv. 51, 2016, str. 1–11.

[24] R. Koh, E. W. Schuster, I. Chackrabarti, A. Bellman, White paper: Securing the pharmaceutical supply chain, 2003.

[25] R. Ganeshan, T. P. Harrison, An introduction to supply chain management, Department of Management Science and Information Systems, Pennsylvania State University, 1995.

[26] Zavod za identifikacijo in elektronsko izmenjavo podatkov – GS1 Slovenija, Uporabniški priročnik GS1, Osnove oštevilčenja in črtnega kodi-

ranja, Dostopno na: http://www.gs1si.org/Standardi/GUM/GUM-17/GS1-UporabniskiPrirocnik-V17.pdf (2016, pridobljeno 2. 9. 2017).

[27] S. Choi, C. Poon, An RFID-based anti-counterfeiting system, zv. 35, 2008, str. 1–12.

[28] C. E. Turcu, C. O. Turcu, M. Cerlinca, T. Cerlinca, R. Prodan, V. Popa, An RFID-based system for product authentication, 2013, str. 32–39.

[29] P. Tuyls, L. Batina, RFID-Tags for Anti-counterfeiting. Topics in Cryptology – CT-RSA 2006, Springer, Berlin, Heidelberg, 2006, str. 115–131.

[30] M. Braun, E. Hess, B. Meyer, Using elliptic curves on RFID tags, zv. 8, 2008, str. 1–9.

[31] N. Druml, M. Menghin, A. Kuleta, C. Steger, R. Weiss, H. Bock, J. Haid, A Flexible and Lightweight ECC-Based Authentication Solution for Resource Constrained Systems, 2014, str. 372–378.

[32] M. Cerlinca, C. Turcu, T. Cerlinca, R. Prodan, V. Popa, Anti-counterfeiting ISO 15693 RFID Solutions Involving Authentication and Traceability Using Symmetric and Asymmetric Cryptography, 2012.

[33] A. J. Menezes, S. A. Vanstone, P. C. V. Oorschot, Handbook of Applied Cryptography, CRC Press, 1996.

[34] D. R. Stinson, Cryptography: Theory and Practice, Third Edition, Chapman & Hall, 2006.

[35] C. P. Pfleeger, S. L. Pfleeger, J. Margulies, Security in Computing (5th Edition), Prentice Hall Press, 2015.

[36] R. C. Merkle, M. E. Hellman, On the security of multiple encryption, zv. 24, 1981, str. 465–467.

[37] E. Barker, A. Roginsky, Transitions: Recommendation for transitioning the use of cryptographic algorithms and key lengths, Special Publication 800-131A, National Institute of Standards and Technology, 2011.

[38] Wikipedia, Hash function, Dostopno na: `https://en.wikipedia.org/wiki/Hash_function` (pridobljeno 23. 9. 2017).

[39] Wikipedia, Cryptographic hash function, Dostopno na: `https://en.wikipedia.org/wiki/Cryptographic_hash_function` (pridobljeno 23. 9. 2017).

[40] J. W. Ralph Jacobi, MIFARE DESFire EV1 AES Authentication with TRF7970A, Dostopno na: `http://www.ti.com/lit/an/sloa213/sloa213.pdf` (2014, pridobljeno 18. 11. 2017).

[41] Gartner.com, Gartner Says Demand for 4G Smartphones in Emerging Markets Spurred Growth in Second Quarter of 2017, Dostopno na: `http://www.gartner.com/newsroom/id/3788963` (pridobljeno 17. 11. 2017).

[42] Android Developers, Android Studio, Dostopno na: `https://developer.android.com/studio/index.html` (pridobljeno 15. 1. 2018).

[43] Microsoft, Visual Studio IDE, Dostopno na: `https://www.visualstudio.com/vs/` (pridobljeno 15. 1. 2018).

[44] Microsoft Azure, What is Azure?, Dostopno na: `https://azure.microsoft.com/en-in/overview/what-is-azure/` (pridobljeno 17. 1. 2018).

[45] MIFARE, TapLinx SDK, Dostopno na: `https://www.mifare.net/en/products/tools/taplinx/` (pridobljeno 18. 11. 2017).

[46] Square, Inc., Retrofit HTTP Library, Dostopno na: `http://square.github.io/retrofit/` (pridobljeno 19. 11. 2017).

[47] Microsoft Docs, ASP.NET Web API, Dostopno na: `https://docs.microsoft.com/en-us/aspnet/web-api/` (pridobljeno 16. 1. 2018).

[48] Microsoft Developer Network, Entity Framework, Dostopno na: https://msdn.microsoft.com/en-us/library/gg696172(v=vs.103).aspx (pridobljeno 15. 1. 2018).

[49] Microsoft Azure, SQL Database, Dostopno na: https://azure.microsoft.com/en-us/services/sql-database/ (pridobljeno 17. 1. 2018).

[50] Microsoft Docs, SQL Server Management Studio, Dostopno na: https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms (pridobljeno 16. 1. 2018).

[51] D. Hardt, The OAuth 2.0 Authorization Framework. RFC 6749. Internet Engineering Task Force, Dostopno na: https://tools.ietf.org/html/rfc6749 (2012, pridobljeno 15. 1. 2018).

[52] E. Hammer-Lahav, The OAuth 1.0 Protocol. RFC 5849. Internet Engineering Task Force, Dostopno na: https://tools.ietf.org/html/rfc5849 (2010, pridobljeno 15. 1. 2018).

[53] S. Ahson, M. Ilyas, RFID handbook: Application, Technology, Security, and Privacy, CRC Press, 2008.

[54] N. Alexiou, S. Basagiannis, S. Petridou, Formal security analysis of near field communication using model checking, zv. 60, 2016, str. 1–14.

[55] T. Hill, R. Westbrook, SWOT analysis: It's time for a product recall, zv. 30, 1997, str. 46–52.